Natural Language Processing

# Assignment Sentiment Analysis

Inholland University of Applied Sciences

15-01-2023
Version I

Tom Denis 670940
Marieke Morganwiese 670515

## Contents

## Introduction

In the dynamic landscape of Natural Language Processing (NLP), sentiment analysis addresses the need to computationally interpret and categorise subjective information embedded in textual data (DeepLearning.AI, 2023). This assignment centres around the 'Sentiment Labelled Sentences' data set, a collection of product reviews where sentences are labelled with their corresponding sentiments. The overarching goal is to leverage NLP techniques to train a classifier capable of predicting sentiment in new sentences accurately. The assignment unfolds in five main experiments (applying lemmatization, Term Frequency-Inverse Document Frequency (Tf.IDF), N-Grams, Bag of Words (BOW) and Global Vectors (GloVe), and Latent Dirichlet Allocation (LDA)), testing the effectiveness of different text preprocessing methods on the Naive Bayes Classifier. Firstly, it emphasises on the development of an effective sentiment classifier through machine learning techniques, with evaluation metrics like accuracy, precision, recall, and F1 score gauging its predictive capabilities. Secondly, it explores the impact of various NLP techniques on sentiment analysis, including lemmatization, tf.idf weights, n-grams, bag-of-word representation, and semantic techniques such as Named Entity Extraction and Topic modelling.

## Preparing the data

Prior to feeding text into a machine learning model for a particular task, it is important to preprocess the text. The preprocessing is carried out to enhance the model's performance and convert words and characters into a format that the model can effectively comprehend (DeepLearning.AI, 2023).

### Exploration

The dataset used in this project comprises three files, each containing labelled text entries where sentences and sentiment labels are paired within individual rows. The dataset contains two types of sentiment, either 0 to denote a negative sentiment of a sentence, or 1 to denote a positive sentiment for a sentence as seen in figure. The graph of total words per sentence distribution can be referenced in Appendix I: Images. All files have been plotted, showing that all text files contain a higher frequency of sentences with little words (0 to 10) than a high number of words (20 to 30).

### Separation

A crucial preprocessing step is to transform this text data into a Pandas data frame with separate columns for the sentence and its sentiment label. Pandas is a Python library, which is used to perform some data exploration, cleaning and analysis (w3schools, 2022). After exploring the text files, it is deduced that the sentiment is always on the last index of its respective sentence, and the sentiment and sentence could be separated on this basis.

### Preprocessing

The main methods applied to clean and prepare the text for further processing are changing the text to lowercase, text cleaning, removal of symbols, and tokenization. Lemmatization is not applied in the preprocessing step, as this is reserved for Experiment 1. Similarly, stemming is not included as this interferes with lemmatization, as the effect of these methods overlap.

The first step in the preprocessing of natural language is to make the text lowercase. Lowercasing is crucial to ensure consistency, as words with different cases contribute the same meaning. Additionally, lowering case aids in reducing vocabulary size, which is beneficial for text vectorization, ultimately improving computational efficiency (Ramya V., 2020). Text cleaning is a process of removing symbols, stop words and punctuation to prepare the text data for further analysis or modelling by eliminating elements that may not contribute significantly to the meaning of the text. Specifically, in the context of stop words, common words like "the," "a," and "an" are removed, as they typically lack specific meaning and do not substantially contribute to the text's understanding (Ramya V., 2020). Tokenization is the process of splitting text into fragments or individual words, called tokens. Each token is an input to the machine learning model which will be used to predict the sentiment of a sentence (DeepLearning.AI, 2023).

## Classification

In the sentiment analysis conducted in this research, the chosen machine learning model is the Naive Bayes classifier. This is a supervised learning algorithm that requires labelled data, under the "naive" assumption of conditional independence between each pair of features, given the class variable's value (Pedregosa et al.,2011).

The experiments begin with establishing a baseline, achieved by running the Naive Bayes model on the pre-processed dataset. The accuracy of this baseline can then be systematically compared to the prediction metrics obtained after each conducted experiment; the precision, recall, and F1-score. Precision measures the percentage of correct positive predictions relative to total positive predictions, recall assesses the percentage of correct positive predictions relative to total actual positives, and the F1-score, being a weighted harmonic mean of precision and recall, provides a comprehensive metric for model quality. The closer the F1-score is to 1, the better the overall model performance (Zach 2022).

In a detailed evaluation through the classification report, the model exhibits balanced performance for sentiment classes 0 and 1. For Class 0, representing a particular sentiment category (e.g., negative sentiment), the precision, recall, and F1-score are all at 81%. This means that 81% of the predictions for negative sentiment are correct, and it captures 81% of the actual negative sentiments present in the dataset. Similarly, for Class 1, representing the opposite sentiment category (e.g., positive sentiment), the model achieves slightly higher values with precision, recall, and F1-score all standing at 82%. This signifies that 82% of the predictions for positive sentiment are accurate, and it captures 82% of the actual positive sentiments in the dataset. The overall accuracy of 81% reflects the model's ability to correctly classify sentiments across both categories.

## Experiments

A total of five experiments are conducted. Each of these tests shows the effect of different language processing methods on the accuracy of the Naive Bayes model. The experiments are applying lemmatization, Term Frequency-Inverse Document Frequency (Tf.IDF), N-Grams, Bag of Words (BOW) and Global Vectors (GloVe), and Latent Dirichlet Allocation (LDA). How these experiments are conducted and what the effect of these applications are on the dataset are discussed below.

1. Lemmatization

Lemmatization is a text processing technique that involves reducing words to their base or root form, known as a lemma. The purpose is to unify different inflected forms of a word to analyse or represent them consistently. This helps in tasks like text analysis, where variations of a word (e.g., "running" and "ran") are treated as the same base form (e.g., "run"). Lemmatization aids in simplifying text for more effective natural language processing and analysis (Gillis, A., 2023).

The lemmatize_words function employs the Natural Language Toolkit (NLTK) library to lemmatize words in the specified column ‘sentences_preprocessed’ of the pandas Data Frame. It begins by creating a WordNet Lemmatizer instance. The function then utilises the apply method on the ‘sentences_preprocessed’ column to iterate through each sentence. For each sentence, it tokenizes the words, assigns part-of-speech tags using the pos_tag function, and lemmatizes each word with the corresponding WordNet POS tag. The resulting lemmatized words are joined back into sentences, and the specified column in the Data Frame is updated with the lemmatized content. This function is designed to enhance text preprocessing by reducing words to their base forms, considering their grammatical roles for accurate lemmatization.

2. Term Frequency-Inverse Document Frequency (TF.IDF)

The second experiment uses TF.IDF to evaluate a term's importance within a collection of documents. This measures how important a word is in a document compared to its frequency across all documents. It emphasises words that are frequent in a specific document but rare in the overall collection, highlighting unique and meaningful terms. By multiplying a term's frequency in a document (TF) with its rarity across the entire dataset (IDF), TF-IDF assigns scores to words, assigning more weight to less frequently used words (Jay S., 2023).

The apply_tfidf_and_classify function is designed for text data preprocessing and sentiment analysis using TF-IDF vectorization and Naive Bayes classification. It leverages the scikit-learn library to create a TF-IDF matrix from the textual content stored in the specified column of the given pandas Data Frame. The data is then split into training and testing sets, and a Multinomial Naive Bayes classifier is trained on the training set. The model's predictions are evaluated using accuracy and a classification report. The function replaces the original text column in the Data Frame with the TF-IDF-transformed data and prints the accuracy and a classification report for model assessment (Das et al, 2018).

3. N-Grams

N-grams are sequential word, symbol, or token sequences within a document, revealing structured relationships between items in the text. Bigrams or trigrams are particularly useful for sentiment analysis, capturing complex expressions like negations (Aashish N. 2021). This experiment focuses on bigrams, enhancing the sentiment analysis capability to consider phrases like 'not good' instead of just 'good' (Raul G., 2024). The apply_ngram_and_classify function incorporates bigrams into sentiment analysis on a pandas Data Frame, using scikit-learn's CountVectorizer. It transforms the 'sentences_preprocessed' feature, checks for existing sparse matrix input, and replaces the original text column with N-grams-transformed data after training a Multinomial Naive Bayes classifier.

### 4.1 Bag of Words (BOW)

The Bag of Words (BoW) model in Natural Language Processing transforms text into a numerical format by representing documents as unordered sets of words, focusing on word frequency rather than grammar or order. Before creating the BoW model, text preprocessing steps such as converting to lowercase, removing non-word characters, and punctuations are essential. The BoW model counts unique word occurrences, selects top words to form a manageable vocabulary, and converts sentences into vectors, resulting in a matrix for machine learning input.  The bow_and_classify function performs sentiment analysis on a DataFrame using BoW, employing CountVectorizer for transformation and training a Multinomial Naive Bayes classifier on the specified column (sentences_preprocessed). The function evaluates performance, replacing the original text column with the BoW-transformed data.

### 4.2 Global Vectors (GloVe)

GloVe, or Global Vectors for Word Representation, is an unsupervised learning algorithm generating word embeddings by constructing a global word-word co-occurrence matrix from a corpus. It captures semantic relationships with linear substructures, offering vector representations of words in continuous space (Japneet S. C. , 2018). The glove_sentiment_analysis function conducts sentiment analysis on a DataFrame using pre-trained GloVe word vectors, loading the model, obtaining average GloVe vectors for pre-processed sentences, and training a Multinomial Naive Bayes classifier. It then evaluates and displays accuracy and a classification report for sentiment predictions on the test set.

### 5. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a potent technique for revealing latent topics in a document corpus, assuming each document is a mix of latent topics, and each word is linked to a specific topic (Pawang, 2021). The apply_lda_and_sentiment_analysis function, leveraging pandas, scikit-learn, and NumPy, seamlessly integrates LDA into sentiment analysis on a Data Frame. It starts with CountVectorizer to transform the 'sentences_preprocessed' column into a bag-of-words representation.The function utilises scikit-learn's LatentDirichletAllocation for topic modelling and MultinomialNB for sentiment analysis, revealing top words for each topic.

## Results

The results of the Naive Bayes classification after each experiment have been applied are discussed below. A summary of these statistics can be found in Appendix II: Table of Accuracy Comparison.

### Lemmatization

The experiment incorporating lemmatization in the Naive Bayes sentiment classification model demonstrated a subtle yet positive impact on accuracy compared to the original baseline. The baseline Naive Bayes model achieved an accuracy of 81.17%, while Experiment 1, enriched with lemmatization, showcased a slightly elevated accuracy of 81.5%. This marginal improvement suggests that the lemmatization preprocessing step contributed to refining the model's ability to correctly classify sentiments within the dataset. While the difference in accuracy is modest, it underscores the effectiveness of linguistic normalisation techniques, such as lemmatization, in enhancing the overall performance of sentiment analysis models on textual data.

### Term Frequency-Inverse Document Frequency (Tf.IDF)

Experiment 2, which introduced the incorporation of TF-IDF weights into the Naive Bayes sentiment classification model, demonstrated a substantial improvement in accuracy compared to the baseline. The accuracy for Experiment 2 reached 82.5%, indicating a noteworthy enhancement in the model's ability to accurately classify sentiments within the dataset. This improvement is particularly significant when compared to the baseline accuracy of 81.17%. The utilisation of TF-IDF, capturing the importance of terms in the context of the entire dataset, proved to be a powerful feature in refining the sentiment classification model's performance.

### N-Grams

Experiment 3, focuses on the integration of N-grams into the Naive Bayes sentiment classification model, displaying a distinct shift in accuracy compared to the baseline and preceding experiments. The model's accuracy for Experiment 3 was notably lower, standing at 62.17%. This represents a substantial decrease when compared to the baseline accuracy of 81.17% and the improved accuracies observed in Experiments 1 and 2. The introduction of N-grams, while capturing additional context, seems to have introduced complexities that affected the model's ability to accurately classify sentiments in this specific dataset, resulting in a notable reduction in overall accuracy.

### Bag of Words (BOW) Versus Global Vectors (GloVe)

In the fourth experiment, the impact of Bag of Words (BOW) and GloVe embeddings on sentiment analysis was explored. Utilising BOW, the Naive Bayes model achieved an accuracy of 81.17%, maintaining precision, recall, and F1-score of 81% for class 0 (negative sentiments) and 82% for class 1 (positive sentiments). This performance served as our baseline. Subsequently, when the model was trained with GloVe embeddings, the accuracy dropped to 74%, with precision, recall, and F1-score of 73% for class 0 and 72% for class 1. Comparing these results, it's evident that the BOW approach outperformed the GloVe embeddings in this sentiment analysis task on the given dataset. This suggests that, in this specific context, BOW representations were more effective than GloVe embeddings for sentiment classification.

### Latent Dirichlet Allocation (LDA)

Experiment 5, focusing on the impact of Latent Dirichlet Allocation (LDA) on sentiment classification, resulted in an accuracy of 81.17%. Comparing this with the baseline accuracy from the experiment set-up, which also stands at 81.17%, there is no noticeable difference in accuracy. Both approaches, with and without LDA, exhibit the same accuracy level. This suggests that, in the context of this specific dataset and sentiment analysis task, the introduction of LDA as a topic modelling technique doesn't significantly alter the Naive Bayes model's accuracy.

### Significance

One-way ANOVA, or Analysis of Variance, is a statistical technique designed to assess whether there are significant differences among the means of three or more independent groups (Zach, 2018). The hypotheses in this case are: **H0:** The experiments have an significant effect on the accuracy of Naive Bayes classification of sentiment. **H1:** The experiments have an insignificant effect on the accuracy of Naive Bayes classification of sentiment. The P-value obtained from the ANOVA test is 0. 623. A P-value close to 1 suggests no difference between the groups other than due to chance (Dahiru, 2008).

## Conclusion

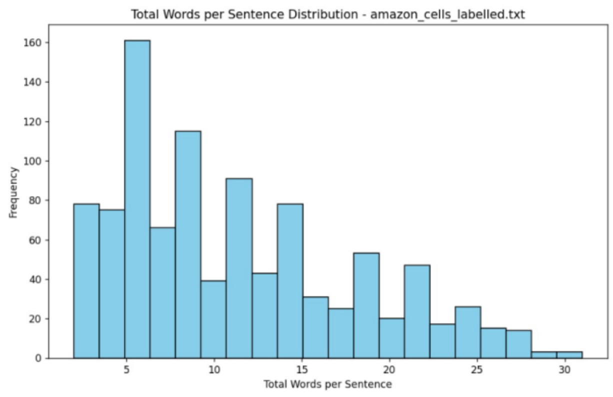
The study explores sentiment analysis in Natural Language Processing (NLP) using the 'Sentiment Labelled Sentences' dataset. The objective was to develop a robust sentiment classifier by applying various NLP techniques systematically. The Naive Bayes classifier served as the baseline, achieving 81% accuracy. Lemmatization improved accuracy to 81.5%, highlighting its significance. Experiment 2 with TF.IDF yielded 82.5% accuracy, emphasising term importance weighting. However, bigrams in Experiment 3 led to a decline to 62.17%, suggesting potential complexity. Experiment 4 showed BoW outperforming GloVe (81.17% vs. 74%). Lastly, LDA had minimal impact, maintaining 81.17% accuracy. These findings provide insights into the nuanced interactions between NLP techniques, showcasing their contributions to sentiment analysis accuracy.

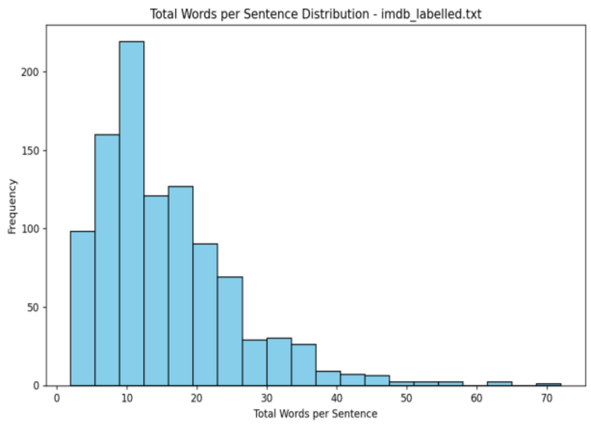
## Discussion

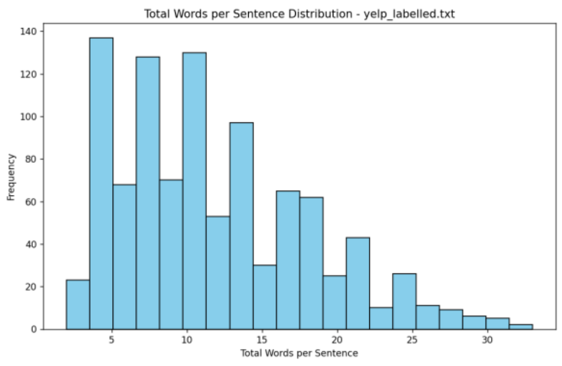
The results from these experiments have been illuminating in their inner workings, but their effects have been negligible. The one-way ANOVA showed that the results before and after the experiments are insignificant with a P-value of 0.62.
The fact that most sentences in the database used are short (0 to 10 words) (Appendix I: Images) can influence these results. A lack of words can diminish the context sentences provided to the machine learning model (Agrawal et al, 2014).
Training sentiment analysis models with predominantly short sentences might result in data sparsity issues. The model may struggle to generalise well if it hasn't seen enough diverse examples, especially those with longer and more complex structures (Parsons, 2008).

A different reason for the relatively low accuracy and little improvement in some experiments is the data size. A bigger data size might have trained the models better and had more material to substantiate the effect of the experiments with. On top of that, hyperparameter tuning was outside the scope of this project. This might have positively impacted the obtained results. Combining two or more experiments together could provide interesting results as well.

## Appendix I: Images







## Appendix II: Table of accuracy comparison

| **Metrics** | **Baseline** | **Experiment 1: Lemmatization** | **Experiment 2: TF.IDF** | **Experiment 3: N-grams** | **Experiment 4: BOW** | **Experiment 4: GLOVE** | **Experiment 5: LDA** |
|---|---|---|---|---|---|---|---|
| Accuracy | 0.81 | 0.82 | 0.83 | 0.62 | 0.81 | 0.74 | 0.81 |
| precision 0 | 0.81 | 0.81 | 0.84 | 0.58 | 0.81 | 0.73 | 0.81 |
| precision 1 | 0.81 | 0.82 | 0.81 | 0.76 | 0.81 | 0.75 | 0.81 |
| f1 score 0 | 0.81 | 0.82 | 0.82 | 0.70 | 0.81 | 0.74 | 0.81 |
| f1 score 1 | 0.81 | 0.81 | 0.83 | 0.49 | 0.81 | 0.74 | 0.81 |

## References

- DeepLearning.AI (January 11th, 2023) Natural Language processing. Retrieved from https://www.deeplearning.ai/resources/natural-language-processing/
- W3schools (2022). Pandas Tutorial. Retrieved from. https://www.w3schools.com/python/pandas/default.asp
- Ramya V. (July 5th, 2020) A  hand book to text Preprocessing. Retrieved from https://towardsdatascience.com/a-handbook-to-text-preprocessing-890f73fd28f8
- Pedregosa *et al.(2011) Scikit-learn: Machine Learning in Python,* 1.9 Naive Bayes .Retrieved from https://scikit-learn.org/stable/modules/naive_bayes.html
- Saumyab271 (13th July, 2023) Stemming vs Lemmatization in NLP: Must-Know Differences. Retrieved from https://www.analyticsvidhya.com/blog/2022/06/stemming-vs-lemmatization-in-nlp-must-know-differences/
- Jay S. (July 31th, 2023) TF-IDF in Sentiment Analysis . Retrieved from https://www.tutorialspoint.com/tf-idf-in-sentiment-analysis
- Nithyashree V(January 12th, 2024).What Are N-Grams and How to Implement Them in Python? Retrieved from https://www.analyticsvidhya.com/blog/2021/09/what-are-n-grams-and-how-to-implement-them-in-python/
- Aashish Nair (November 1st, 2021).Leveraging N-grams to Extract Context From Text .Retrieved from https://towardsdatascience.com/leveraging-n-grams-to-extract-context-from-text-bdc576b47049
- Zach (December 27th, 2018 ) One-Way ANOVA: Definition, Formula, and Example. Retrieved from https://www.statology.org/one-way-anova/#:~:text=A%20one-way%20ANOVA%20%28%E2%80%9Canalysis%20of%20variance%E2%80%9D%29%20compares%20the,statistically%20significant%20difference%20between%20the%20corresponding%20population%20means
- Raul Garreta (January 5th, 2024) N-gram range. Retrieved from https://help.monkeylearn.com/en/articles/2174105-n-gram-range
- Noob_coders_ (March 8th, 2019) , Bag of words (BoW) model in NLP. Retrieved from https://www.geeksforgeeks.org/bag-of-words-bow-model-in-nlp/
- Japneet Singh Chawla, (April 24th, 2018) What is GloVe? .Retrieved from https://medium.com/analytics-vidhya/word-vectorization-using-glove-76919685ee0b#:~:text=GloVe%20stands%20for%20global%20vectors%20for%20word%20representation.,linear%20substructures%20of%20the%20word%20in%20vector%20space
- Pawang (June 6th ,2021) Latent Dirichlet Allocation. Retrieved from https://www.geeksforgeeks.org/latent-dirichlet-allocation/
- Bijoyan Dash et al (2018), An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation, retrieved from https://arxiv.org/ftp/arxiv/papers/1806/1806.06407.pdf
- Alexander S. Gillis (20 October 2023),  lemmatization, retrieved from https://www.techtarget.com/searchenterpriseai/definition/lemmatization#:~:text=Lemmatization%20is%20the%20process%20of,processing%20(NLP)%20and%20chatbots

- https://www.techtarget.com/searchenterpriseai/definition/lemmatization#:~:text=Lemmatization%20is%20the%20process%20of,processing%20(NLP)%20and%20chatbots
- Tukur Dahiru (6 June 2008). Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4111019/#:~:text=The%20P%20value%20is%20defined,groups%20is%20due%20to%20chance
- Agrawal et al (August 23 2014). Retrieved from https://aclanthology.org/S14-2065.pdf
- Jeffrey Parsons (April 2008). Retrieved from (https://www.researchgate.net/publication/345668285_A_Sentence-Level_Sparse_Gamma_Topic_Model_for_Sentiment_Analysis