Sentiment Analysis with Bag-of-Words

Romain BLANCHOT 67540460063

May 18, 2025

Sentiment Analysis with Bag-of-Words — Report

1. Introduction

This project was part of a hands-on NLP project, with a particular focus on sentiment analysis. The main objective was to implement a complete NLP pipeline using the Bag-of-Words (BoW) approach, as requested in the exercise. To enrich my understanding, I also investigated possible alternatives such as TF-IDF and Transformer-based models.

The Amazon Reviews dataset was a natural choice. Unlike analyzing tweets, which had no direct professional application for me, Amazon customer reviews represented a concrete use case. This choice proved particularly relevant as it met a real need: the analysis of 500 customer reviews in a hotel context. Indeed, this project will serve as the basis for a future professional application as part of my freelance activity, where I plan to adapt the model by replacing BoW with more sophisticated approaches such as FastText or DistilBERT.

This practical experience has enabled me to understand not only the technical aspects of implementing an NLP pipeline, but also the importance of choosing data and models according to the context of use. The planned transition to more advanced models illustrates the natural evolution of NLP solutions, from the traditional BoW approach to more modern, high-performance architectures.

I highly recommend to see de git repository of the project here.

2. Technical Pipeline

1. **Loading and parsing the `.bz2` file**
   - Using `kagglehub` to download the dataset
   - Reading 10,000 lines from `train.ft.txt.bz2`
   - Parsing FastText labels (`__label__1/2`) and text
2. **Preprocessing**
   - Removing punctuation
   - Converting to lowercase
   - Lemmatization with NLTK WordNetLemmatizer
   - Removing English stopwords
   - Extracting 1-grams and 2-grams
3. **BoW Vectorization**
   - Using `CountVectorizer`
   - Limiting to 3,000 features
   - Parameters: `max_df=0.95`, `min_df=2`

- Sparse matrix of dimension (10000, 3000)

4. **Train/test split**
   - 80/20 split with `train_test_split`
   - Fixed random state for reproducibility
5. **Tested models**
   - LogisticRegression (max_iter=1000)
   - MultinomialNB
   - SGDClassifier (max_iter=1000)
6. **Evaluation**
   - Metrics: accuracy, precision, recall, F1-score
   - Support for each class
   - Evaluation on 2,000 test samples

3. Results and Discussion

Model Results

Logistic Regression **Accuracy**: 0.8300

Class	Precision	Recall	F1-score	Support
1	0.83	0.84	0.84	1037
2	0.83	0.82	0.82	963

Global metrics: - Accuracy: 0.83 - Macro avg: 0.83 - Weighted avg: 0.83

Multinomial Naive Bayes **Accuracy**: 0.8245

Class	Precision	Recall	F1-score	Support
1	0.83	0.83	0.83	1037
2	0.81	0.82	0.82	963

Global metrics: - Accuracy: 0.82 - Macro avg: 0.82 - Weighted avg: 0.82

SGD Classifier **Accuracy**: 0.8105

Class	Precision	Recall	F1-score	Support
1	0.81	0.82	0.82	1037
2	0.81	0.80	0.80	963

Global metrics: - Accuracy: 0.81 - Macro avg: 0.81 - Weighted avg: 0.81

Results Analysis

The results show very similar performance between the three tested models, with a maximum difference of only 2 percentage points. Logistic Regression stands out slightly with 83% accuracy, closely followed by Multinomial Naive Bayes (82.45%) and SGD Classifier (81.05%).

Key points: - All models show excellent balance between precision and recall - Class 1 (positive) is slightly better predicted than class 2 (negative) - Global metrics (macro and weighted averages) are very stable - Support is balanced between classes (1037 vs 963 samples)

These results confirm the robustness of the BoW approach for sentiment classification, even with relatively simple models. The 83% accuracy performance is particularly satisfactory for a first implementation, paving the way for future improvements with more sophisticated architectures.

BoW Strengths and Weaknesses

Strengths: - Simple implementation - Fast training - Good performance on binary classification tasks - Captures n-grams for local context

Weaknesses: - Loss of word order - No semantic context consideration - Vocabulary sensitivity - Limited by number of features

Parameter Impact

- `max_features=3000`: good compromise between performance and memory
- `ngram_range=(1,2)`: important local context capture
- `max_df=0.95`: elimination of too frequent terms
- `min_df=2`: removal of rare terms

4. Problems Encountered & Solutions

Problem	Solution
FastText Format	Adapted parser to handle `__label__1/2` labels
NLP vs ML Pipeline	Added specific steps (lemmatization, n-grams)
Model Choice	Tested several classifiers (LogReg, MultinomialNB, SGD)
BoW Memory Size	Used sparse matrices `scipy.sparse.csr_matrix`

5. Conclusion & Improvement Paths

- Use POS tagging
- Replace BoW with TF-IDF
- FastText / DistilBERT for more context
- Grid-search hyperparameters