# Movie Sentiment Analysis: A Multinomial Naïve Bayes-Based Approach for Assessing User and Critic Opinions

Yassine Rodani
University of Haute-Alsace, FR
yassine.rodani@uha.fr

## Abstract

*The growing volume of user-generated content, particularly in the form of movie reviews, presents both challenges and opportunities for researchers and practitioners in the movie industry. Sentiment analysis (SA) has emerged as a vital tool to automatically process and interpret this vast data repository, offering valuable insights into viewer preferences and opinions [1]. This paper presents a comprehensive review of the techniques and applications of movie sentiment analysis, highlighting the role of natural language processing and machine learning algorithms in extracting sentiment from text data. In this project, I involve a comprehensive study of relevant NLP techniques, including data pre-processing, feature extraction, and model selection. The chosen Multinomial Naïve Bayes algorithm will be trained on a data set of user critic reviews, with model performance evaluated based on multiple evaluation metrics. Results demonstrate a high classification accuracy of 86.3%, which indicates the effectiveness of the proposed solution. This confirms the potential of incorporating the designed approach into modern text-based sentiment analysis tools.*

## 1. Introduction

In recent years, the movie industry has experienced exponential growth, with an increasing number of films being produced and consumed worldwide [2]. Understanding audience and critic opinions has become crucial for filmmakers, marketers, and distributors to create and promote content that resonates with viewers. Traditionally, the analysis of movie reviews has relied on manual methods, which can be time-consuming and prone to subjectivity. As a result, there is a pressing need for an automated solution that can efficiently and accurately analyze movie sentiment. [3]

The Movie Sentiment Analysis project addresses this need by developing a natural language processing model capable of extracting sentiment from user critic reviews. This project utilizes the Multinomial Naïve Bayes algorithm to predict positive or negative sentiments from text data. The choice of this algorithm is motivated by its simplicity, efficiency, and proven success in text classification tasks.

To ensure the effectiveness of the model, a thorough exploration of relevant NLP techniques will be undertaken, including data preprocessing, feature extraction, and model optimization. A comprehensive evaluation of the model's performance will be carried out, using metrics such as accuracy, F1 score, and a confusion matrix. Ultimately, the Movie Sentiment Analysis project will be deployed as a user-friendly Python web application using the Streamlit framework, which will be containerized with Docker. This approach allows users to easily access and utilize the system.

The paper is structured as follows: Section 2 describes the related work in the area of SA. Section 3 covers the materials and methods used in development. Section 4 outlines the proposed methodology. Section 5 presents the results of the proposed approach. Section 6 delves into the deployment of the model, while Section 7 concludes the research and explores future work.

## 2. Related Work

Over the years, sentiment analysis has evolved as a significant research area, enabling the extraction of meaningful information from large volumes of user-generated content. Many studies have focused on developing techniques and models to effectively analyze sentiments in various domains, including product reviews, social media posts, and movie reviews [3, 4]. In the context of movie sentiment analysis, researchers have employed various natural language processing (NLP) and machine learning algorithms to better understand viewer preferences and opinions [5, 6].

One of the earliest works in sentiment analysis, by Turney [5], focused on using unsupervised learning to classify movie reviews as positive or negative. Further advancements in the field led to the exploration of supervised learning techniques, such as Naïve Bayes, Support Vector Machines, and deep learning models like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks [6–8]. These models have demonstrated varying degrees of success in capturing the nuances of sentiment in movie reviews.

Recent studies have also explored the use of pre-trained language models, such as BERT and GPT, to improve sentiment analysis performance by leveraging their ability to understand complex linguistic patterns [9, 10]. In addition to advancements in modeling techniques, researchers have focused on refining data pre-processing and feature extraction methods, such as tokenization, stemming, and the use of word embeddings, to better represent the underlying sentiment in the text [11, 12].

## 3. Materials and Methods

### 3.1. Data

For this project, I will be using a data set containing 50k movie reviews from IMDb [13]. The data have already been splitted into 25k reviews for training purposes while the other 25k is intended for testing the classifier. In addition, both sets contain 12.5k positive and negative reviews as shown in Figure 1.

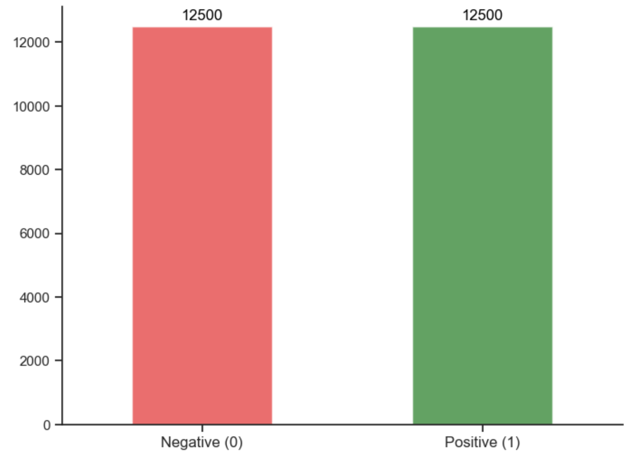

Figure 1. Class distribution of the IMDb movie reviews data set [13], illustrating an equal split of 12.5k positive and negative reviews in both the training and testing sets. This balanced distribution helps ensure fair evaluation and generalization of the classifier.

The reviews are classified into positive and negative in reference to the IMDb rating system. It allows viewers to rate on a scale from 1 to 10, and according to the data set creator anything less than 4 stars is labeled negative, and above 7 stars is marked as positive. Reviews with ratings out of the above ranges are not included. There are at most 30 reviews for each movie. The average number of words per review is 233.79 with a standard deviation of 173.73 words as shown in Figure 2.

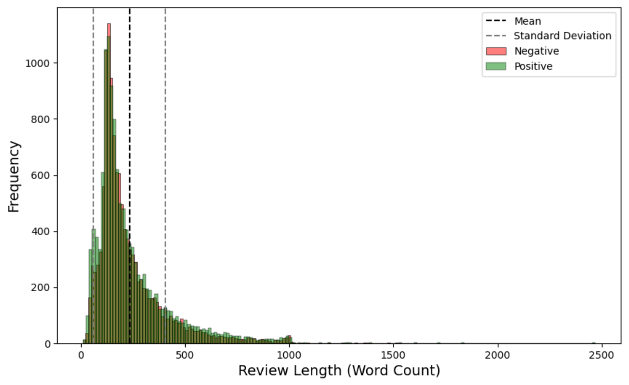

Figure 2. Distribution of Review Length by Sentiment (average: 233.79, standard deviation: 173.73), showing the frequency of movie reviews split by sentiment (positive or negative) and the corresponding distribution of review lengths (measured in word count). The black dashed line indicates the mean review length, while the gray dashed lines indicate one standard deviation above and below the mean.

The spread is similar in shape for both types of reviews however negative reviews are on average a tad shorter.

### 3.2. Preprocessing

For any machine learning project, once you've gathered the data, the first step is to process it to make useful inputs to your model. Data preprocessing is a crucial task in the data mining process. It refers to cleaning up the data from useless information that will not help in the training process and might cause confusion during the classification process. For the IMDb data set, several data preprocessing steps are utilized.

- **Text Tokenization:** As illustrated in Figure 3, tokenization is the process of dissecting a text segment into smaller units, referred to as tokens. These tokens can consist of words, word fragments, or punctuation marks, allowing for more effective natural language processing and analysis.

- **Word Filtering:** To remove noise from the data, we will first eliminate words that don't provide much information about the content, such as common words like 'I, you, are, is, etc.' that don't offer sufficient insight into sentiment. Then, we will remove hyperlinks, hashtags, and punctuation to ensure words with or without punctuation are treated as the same word. For example, "happy", "happy?", "happy!", "happy," and "happy." should all be considered as the same word.

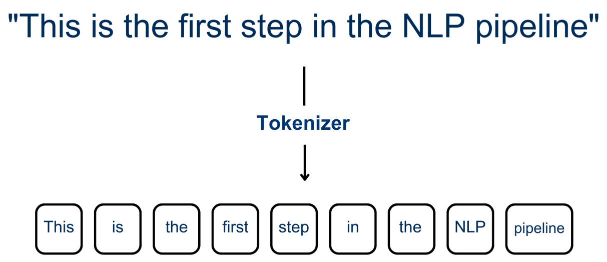

Figure 3. Visual representation of the tokenization process, showcasing the transformation of raw text input into individual tokens.

- **Stemming:** Stemming entails eliminating prefixes and suffixes from a word to obtain its base or root form. The Porter stemming algorithm, a rule-based method developed by Martin Porter [14], is commonly employed for this purpose. In this algorithm, a consonant is defined as any letter excluding vowels. The calculation of conditions in the algorithm is demonstrated in Form 1, where optional content is represented by square brackets, and $(VC)^m$ indicates a sequence of a Vowel $(V)$ followed by a Consonant $(C)$ repeated $m$ times.

$$[C]\,(VC)^m\,[V] \tag{1}$$

  This algorithm adheres to a set of rules that consist of patterns along with their associated conditions, the rules follow the Form 2:

$$S1 \longrightarrow S2 \tag{2}$$

  If a pattern is found to match and the word concludes with the $S1$ suffix, the suffix is converted from S1 to $S2$, and the algorithm starts anew from the list's beginning to identify the subsequent matching pattern. Should no pattern match, the algorithm then outputs the result.

### 3.3. Word Embedding

Before training the model, we need to transform our cleaned reviews into numerical values so that the model can understand the data. This process is what we call Word Embedding. By mapping words into a high-dimensional vector space, word embeddings capture the relationships between words, allowing for the identification of synonyms, antonyms, and other linguistic patterns. In this project, I will be using TfidfVectorizer method from scikit-learn [15] that will help to convert a collection of text documents to a matrix of TF-IDF features.

### 3.4. Multinomial Naïve Bayes Classifier

A Multinomial Naïve Bayes classifier is a probability-based machine learning model utilized for classification tasks [16]. This classifier fundamentally relies on the principles of Bayes' theorem as shown in Equation 3.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3}$$

By applying Bayes' theorem, we can determine the probability of event A occurring, given that event B has already taken place. In this context, B represents the evidence and A stands for the hypothesis. The underlying assumption is that the predictors or features are independent, meaning the presence of one feature does not influence the others. This is why the classifier is deemed ”naive”.

This classifier is commonly used for document classification tasks, such as determining the category a document belongs to, be it sports, politics, technology, or others. The features or predictors employed by the classifier are based on the frequency of words found within the document.

In the context of the movie sentiment analysis project, the Multinomial Naive Bayes classifier operates under these assumptions to classify movie reviews as positive or negative. We would focus on counting the number of positive and negative words or phrases present within the text. This can be achieved by creating a predefined list of positive and negative words, often referred to as a ”sentiment lexicon”. The classifier then calculates the frequency of these words within the document, allowing it to determine the overall sentiment of the text. For example, a higher frequency of positive words would suggest a positive sentiment, whereas a higher frequency of negative words would suggest a negative sentiment.

## 4. Proposed Approach

In this project, I propose a sentiment analysis model using Multinominal Naïve Bayes algorithm. The execution steps that are shown in Figure 4 could be summarized as the following:

1. Raw Text
2. Text Tokenization: String to Word Vector
3. Removing unwanted words using regular expressions
4. Removing unwanted words manually
5. Stemming
6. Word Embedding using TF-IDF method
7. Multinomial Naïve Bayes Classifier
8. Sentiment Predictions

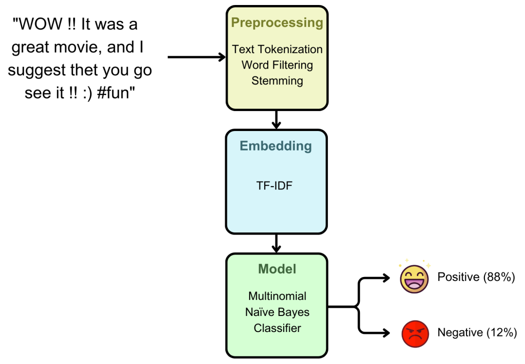
"WOW !! It was a great movie, and I suggest thet you go see it !! :) #fun"
Preprocessing
Text Tokenization
Word Filtering
Stemming
Embedding
TF-IDF
Model
Multinomial Naïve Bayes Classifier
Positive (88%)
Negative (12%)

Figure 4. The proposed system flowchart

## 5. Results

For evaluation, three metrics were used: accuracy, f1-score, and a confusion matrix to provide a tabular visualization of the ground-truth labels versus model predictions as shown in Figure 5. The two first measures can be calculated by applying Equations 4, 5, 6, and 7, where $TP$ stands for True Positives, $TN$ stands for True Negatives, $FP$ stands for False Positives, and $FN$ stands for False Negatives.

### Accuracy

Accuracy calculates the ratio of correctly predicted instances to the total instances in the data set. We can use Accuracy when we have a balanced data set (i.e., similar proportions of each class) and when both false positives and false negatives are of equal importance. The accuracy score is found to be around 86.3% on the testing set.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

### F1-Score

F1-Score is a measure combining both precision and recall. It is generally described as the harmonic mean of the two. Harmonic mean is just another way to calculate an "average" of values, generally described as more suitable for ratios (such as precision and recall) than the traditional arithmetic mean.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F_1 = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{7}$$

A classification report as shown in Table 1 is provided to summarize all these measures.

Table 1. Classification report

| | precision | recall | f1-score |
|---|---|---|---|
| class 0 | 0.85 | 0.88 | 0.87 |
| class 1 | 0.87 | 0.85 | 0.86 |
| macro avg | 0.86 | 0.86 | 0.86 |

### Confusion Matrix

Confusion matrix provides a summary of correct and incorrect predictions. The results are summarized and presented in Figure 5.

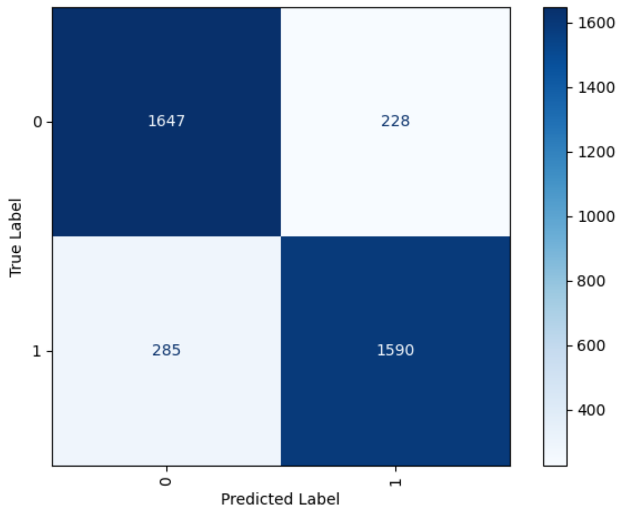

Figure 5. Confusion matrix illustrating the true positive, true negative, false positive, and false negative rates for the classifier's performance on the testing set.

Considering both the accuracy score of the classifier and the confusion matrix, the results demonstrate that the classifier achieves satisfactory precision on the IMDb dataset, effectively balancing true positive and negative predictions.

## 6. Model Deployment

The deployment process is streamlined and efficient, utilizing modern technologies such as Streamlit and Docker. The process can be visualized in Figure 6, which illustrates the steps involved: creating the Streamlit application, building a Docker image using a Dockerfile, pushing the image to a container registry, pulling the image, and finally running the Docker container, which hosts our application. This deployment strategy ensures a consistent and easily maintainable application environment, enabling seamless integration with various platforms and systems. By containerizing the application with Docker, we can harness the full potential of our sentiment analysis tool, enhancing the overall understanding of viewer preferences and opinions.

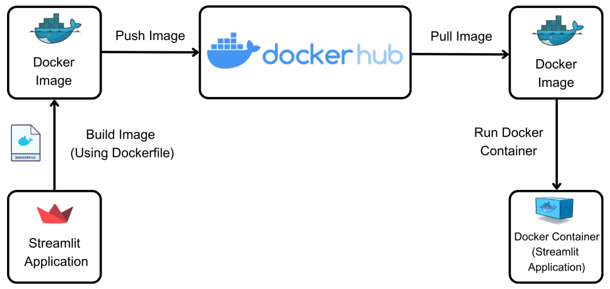

Figure 6. An illustration of the model deployment process for the Sentiment Analysis application, showcasing the integration of Streamlit and Docker for a seamless, maintainable, and scalable deployment.

## 7. Conclusion

Sentiment Analysis also referred to as opinion mining, involves extracting opinions from text data and classifying them into positive, negative, or neutral ones. In this project, a Multinomial Naïve Bayes classifier is used to automatically categorize the preprocessed IMDb movie reviews. In total, 25k reviews are considered, 12.5k for positive and 12.5k for negative sentiments. Results have concluded that the highest accuracy attained by the devised approach is 86.3%. A superior accuracy can be attained by using further data preprocessing techniques. Furthermore, higher classification accuracy can be achieved by employing other classifiers or deep learning approaches. Exploring these opportunities is another prospect.

## Code Implementation

The GitHub repository of this project is made publicly available on: https://github.com/yassine-rd/movie-sentiment-analysis.

## Acknowledgement

I would like to thank the authors [13] for creating and providing the dataset used in this project. Their contribution has been invaluable in enabling me to conduct my analysis and draw meaningful insights.

## References

[1] A. Wijesinghe, "Sentiment analysis on movie reviews," 10 2015. 1

[2] Statista, "Global film industry growth 2019-2025," https://www.statista.com/statistics/259987/film-industry-revenue-growth-worldwide/, 2021. 1

[3] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008. 1

[4] B. Liu, *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers, 2012. 1

[5] P. D. Turney, "Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews," *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 417–424, 2002. 1

[6] P. Lohar, H. Afli, and A. Way, "A comparative study of supervised machine learning algorithms for sentiment classification," in *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, 2018. 1, 2

[7] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002, pp. 79–86. 2

[8] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, 2011, pp. 142–150. 2

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2019. 2

[10] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018. 2

[11] D. Jurafsky and J. H. Martin, *Speech and language processing*, 3rd ed. Prentice Hall, 2019. 2

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013. 2

[13] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. USA: Association for Computational Linguistics, 2011, p. 142–150. 2, 5

[14] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 40, pp. 211–218, 1997. 3

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. 3

[16] Vikramkumar, V. B, and Trilochan, "Bayes and naive bayes classifier," 2014. 3