# Analyzing the Strengths and Limitations of Naïve Bayes in Text Classification

Goilean Giulia Carla Alexa
**Advanced Methods in Data Analysis – Report 1**
Group: 246/1
giulia.goilean@stud.ubbcluj.ro

## Abstract

This study examines the strengths and limitations of Naïve Bayes classifier for text classification for natural language processing tasks. It uses three different datasets that increase gradually in difficulty: Restaurant Reviews, News Topics and IMDb Movie Reviews. The analysis evaluates how preprocessing choices, smoothing, text representations or feature configurations can influence a model's performance. A common experimental setup was applied to all datasets: incorporating tokenization, stop word filtering, lemmatization and two variants of vectorization: Count and TF-IDF. The study aims to highlight which factors can create a great scenario for the model and understand where its weaknesses are. The results show that Naïve Bayes performs well on sentiment-focused and short texts. However, it struggles in modeling semantic complexity, as it is not well suited with its independence assumption of features.

## 1. Introduction

In the past few years artificial intelligence has become a mainstream subject. Its fast paced evolution has sparked interest and debates in particular about the extent to which machines can understand and process human language. Nowadays, digital systems that feature tasks as automatic content filtering or topic classification are widely used and easily available to the public. Evaluating how well algorithms handle real human language has become a practical and scientific challenge due to the complexity of the subject and the society's curiosity.

Within Natural Language Processing (NLP), Naïve Bayes classifiers remain an important baseline for text classification and a subject of interest. It is characterized by efficiency, simplicity and an unexpected strong performance on large numbers of features [MS99]. Even though Naïve Bayes relies on the unrealistic assumption that words are independent in text, research has proven its success across a wide range of tasks. It has remained a subject of active interest in scientific work, being a foundation for innovation (e.g., Censored Naïve Bayes [Wol14]).

The goal of this study is to understand and experiment with the strengths and limitations of Naïve Bayes in understanding human language across multiple real-world text classification problems. This study will examine three datasets of increasing complexity. Restaurant reviews sentiment analysis contains text reviews from customers with the challenge of classifying if the review is positive or negative, which can be considered a baseline NLP problem. News Topic Classification is a large-scale, diverse dataset involving four news categories. This can test Naïve Bayes ability to perform on information dense text. The third dataset is IMDb Movie Reviews Genres Description and Emotions. It is a multi-class problem testing the ability to understand emotions from fine nuances in text. The experiments follow a systematic methodology: text preprocessing (tokenization, stop word removal, lemmatization), feature extraction using TF-IDF and Bag of Words. For each dataset, the report evaluates and compares the performance of the model, providing insight into how different aspects of text representation affects the behavior of Naïve Bayes.

## 2. Theoretical Background

### 2.1. Naive Bayes Classifiers

In the domain of text categorization one of the most frequently used methods is represented by the Naïve Bayes classifiers, as a probabilistic approach. Characteristics like mathematical simplicity, competitive performance on high-dimensional data such as natural language text (NLP) and computational efficiency have built their popularity. Naïve Bayes belongs to the family of generative models, which means that it explicitly estimates the joint probability distribution P(x,y) and using Bayes' decision rule it assigns a class to a new instance [MS99]. In classification tasks, the goal is to choose the class y that maximizes the posterior probability given the observed features x. The posterior probability is computed using Bayes' rule seen in Figure 1. This approach comes from the core assumption of the model: conditional independence of the features given the class. Therefore, once the category is known, each feature $x_i$ is treated as if it occurs independently of all others. This assumption is rarely true in natural language, as words tend to acquire different meanings in different contexts. However, the literature highlights that the independence assumption simplifies the model and, in practice, does not prevent it from achieving strong predictive performance [Ras14].

$$P(s_k|c) = \frac{P(c|s_k)}{P(c)} P(s_k)$$

**Figure 1.** *Bayes' rule [MS99]*

In text classification, features usually correspond to words or n-grams extracted from the document. The Naïve Bayes classifier evaluates how likely a word is to appear in a text belonging to a particular class and compares it to the evidence across all observed terms. Given that raw language has large vocabularies and sparse word frequencies, Naïve Bayes is suited to Bag-of-Words and TF-IDF representations. They transform variable-length text into fixes-length numerical feature vectors.

There are two main variants of Naïve Bayes used in NLP: Multinomial Naïve Bayes and Bernoulli Naïve Bayes [Ras14]. The first approach is based on word counts and term frequency. It treats each document as a sequence of token occurrences and estimates class conditional word distributions. The second one models binary word presence and is suited when the frequency of a word is less informative than its simple appearance.

A possible problem with the Naïve Bayes classifier would be when in training a class receives zero probability, making entire documents impossible to classify. To avoid this situation, a critical factor is smoothing [MS99], especially Laplace smoothing. This allows the model to generalize beyond the observed vocabulary.

In spite of its simplicity, Naïve Bayes performs surprisingly well in real-world NLP applications, including the topics choose for this report's experiments.

### 2.2. Text Representation

Machine learning models cannot directly use textual data, as raw language is unstructured and varies in length. Therefore, transforming documents into numerical feature vectors is an essential step in text classification. This subchapter introduces the main representation methods used during this study: Bag-of-Words (BoW), TF-IDF and n-grams.

The Bag-of-Words model sees each text as a multiset of its words, ignoring order, grammar or syntactic structure. Each document is mapped to a vector of term frequencies. This representation is simple, but effective on many classification tasks because they depend more on lexical cues than on surface syntax [MS99].

Term Frequency-Inverse Document Frequency (TF-IDF) extends BoW by down-weighting common words (e.g., "is", "said", "the") and up-weighting informative ones. This often leads to better performance than raw counts when dealing with large vocabularies, however it can cause problems when the query does not match the words used by the writer in the document [JM23]. In figure 2 $tf_{t,d}$ tells how often the word occurs in the document and $idf_t$ the number of documents in which t occurs.

$$\text{tf-idf}(t,d) = \text{tf}_{t,d} \cdot \text{idf}_t$$

**Figure 2.** *TF-IDF formula [JM23]*

While BoW and TF-IDF rely on unigrams (single words), n-grams generalize the representation of n tokens (words). In this way, the model can capture fixed expressions or negations. On the other hand, increasing n leads to exponential vocabulary growth, so it needs to be used prudently [MS99].

These representations are well suited for Naïve Bayes classifiers for many reasons, especially because they treat features as independent as does Naïve Bayes.

### 2.3. Text Preprocessing for Naïve Bayes

Preprocessing is an important step for probabilistic text classifiers such as Naïve Bayes because the model relies on the statistical properties of word features. Tokenization represents the process of breaking down the text into individual world units. Simple tokenizers are typically suitable for Naïve Bayes because the model does not depend on word order. Another important step is stop words removal. They are common words seen as rather un-informative (e.g., or, the, so) [Ras14]. Another procedure is lemmatization. It aims to map different morphological variants of a word to a single base form. Overall, text preprocessing aims to shape the feature distribution that Naïve Bayes learns from. Those steps help the classification accuracy by reducing noise and keeping only the informative words.

## 3. Proposed procedure

### 3.1. Datasets

**Restaurant reviews** is a tabular dataset containing 2220 rows and 2 columns: Review and Liked. The first column represents textual reviews written by customers. It describes their experience at the restaurant, including positive or negative comments about food, service, cleanliness or overall feedback. The Liked column is a binary label representing the rating given by the customer, 1.0 indicating a positive opinion and 0.0 a negative one. The reviews are short: averaging 8 words per review and the distribution between positive and negative reviews is balanced as seen in Figure 3. This dataset was chosen as the simple scenario in the study. It is a typical short text sentiment classification for Naïve Bayes, providing a great way to analyze and understand the basic mechanics of Naïve Bayes.

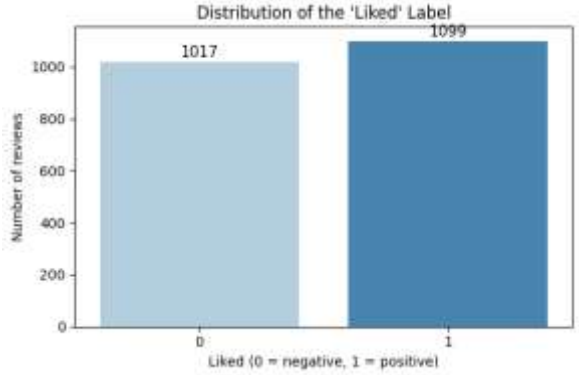



**Figure 3.** *Distribution of Liked label in Restaurant Reviews dataset*

**News Topic Classification** dataset [AGC] contains 120000 training samples of new headlines and descriptions, labeled into one topic categories. The four categories are World(0), Sports(1), Business(2), and Sci/Tech(3). The dataset is perfectly balanced with 30000 samples per class. The average length of a headline is around 38 words, with a more diverse vocabulary and specific terminology for each domain. This dataset was selected to evaluate how Naïve Bayes handles topic classification. In comparison with the Restaurant Review dataset, the challenge for the model is to learn from longer and complex text, where the discriminative cues could be nouns and proper nouns (as seen in Figure 4) rather than sentiment markers.


**Figure 4.** *Word clouds News Topic categories*

**IMDb Movie Reviews** is a dataset that includes movie reviews along a short textual description, a list of genres for each movie and an emotion label. The original dataset includes over 46000 rows. For this study, the Description, genres and emotion columns become the focus as the model tries to assign an emotion for each movie. The emotion label has 8 possible values: sadness, joy, anticipation, optimism, anger, fear, disgust and surprise. A movie can have more than one genre and the average length for descriptions is 27 words. The emotion distribution is highly imbalanced as seen in Figure 5, where the label surprised appears only in 57 instances. Because of that the rows containing that label were removed. This creates a challenge for the model: it has to solve a multi-class, imbalanced emotion classification task on relatively short descriptions.

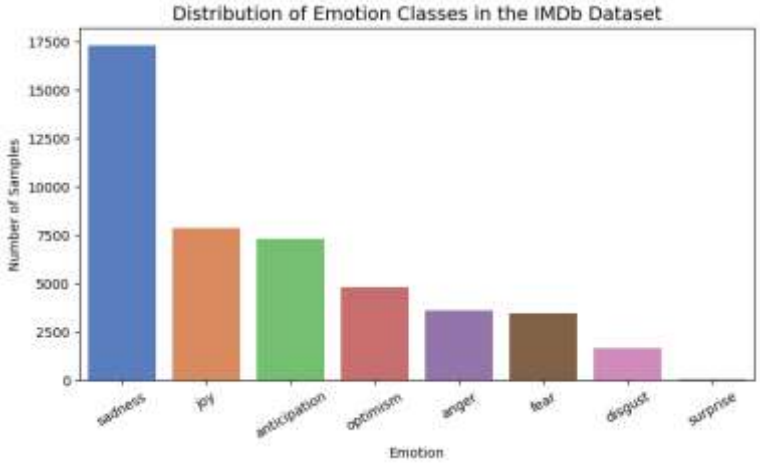



**Figure 5.** *IMDb distribution of emotion classes*

### 3.2. Text Preprocessing

For machine learning models like Naïve Bayes, an essential step in preparing the data is text preprocessing. The algorithm learns statistical patterns for the words and assumes conditional independence between them, so the consistency and quality of the text representation highly influences the model's performance. For all three datasets were chosen similar preprocessing steps.

Firstly, the datasets were analyzed for any missing data, duplicates or inconsistencies. Any faulty rows were removed to prevent overfitting or misleading estimations. From Restaurant Reviews dataset 103 rows were eliminated due to the fact that the Liked value was missing. The IMDb Movie Review dataset needed the most cleaning, as it contained many duplicates of the same movie name and the surprise class was removed as it appeared only in 57 instances. This way, the probability of having an unstable classifier was reduced. From only 57 instances, the model would not have the ability to learn any significant information in order to correctly classify that label.

All textual inputs were standardized through a uniform cleaning process: lowercase transformation and only alphabetic characters were kept. This steps reduced vocabulary sparsity and created a consistent representation. Moreover, stop word removal was done, but negation words ("not", "no", "never") were kept because in many instances they can provide important cues. In reviews or descriptions people tend to use negations to accentuate their experience. Then, from the "nltk" library WordNetLemmatizer was used. Lemmatization reduces words to their canonical dictionary form, This process was chosen as it is helpful for Naïve Bayes: it reduces vocabulary size and prevents feature fragmentation. It was applied to all datasets. Following this approach, the model can see essential descriptors (e.g., "terrified", "terrifying") as one unique feature.

### 3.3. Feature Representation

To covert raw text into numerical features suitable for Naïve Bayes models, the study used the scikit-learn library for two vectorization techniques: TfidfVectorizer and CountVectorizer. CountVectorizer transform text into sparse matrix of raw term counts. It is ideal for topic classification where categories differ by nouns or named entities and it sees high-frequency words as informative. TF-IDF Vectorizer tries to find rare terms that could be discriminative and relevant for the classifier. Both techniques were applied to all datasets to compare and contrast their influence and effect on performance. This study also experiments with n-grams, comparing unigrams and bigrams for every dataset. They capture word sequence of length n, improving the model's ability to understand contextual cues. Certain datasets could benefit from bigrams use, so this study explores their influence. Each model uses a capped vocabulary: Restaurant Reviews uses 1500 features as it is a smaller dataset, News Topic and IMDb reviews use 5000 features to keep relevant descriptive details.

### 3.4. Model Configuration

The Naïve Bayes algorithm used was from the sklearn library. Two variants were considered: MultinomialNB and BernoulliNB. The BernoulliNB variant was used only on the Restaurant Reviews dataset. Because it classifies by the presence or absence of a term it could be useful in the context of a positive or negative feedback, however term frequency carries crucial information for emotion or topic classification. The MultinomialNB was chose for all three datasets, as it handles longer documents and large vocabularies efficiently. This variant is recommended for standard text classification [Ras14].

The Naïve Bayes model estimates the probability of a class by multiplying conditional probabilities. If a word does not appear in a class the probability becomes zero. This can cause problems, as the model multiplies probabilities across all words. To avoid this, a smoothing parameter is used. Laplace or Lidstone smoothing adjusts probability estimates so that no term has zero probability. This adds a pseudo-count $\alpha$ to each term. The default value $\alpha = 1$ was chosen because it offers both stability and sensitivity. Additional tests with different values for $\alpha$ where performed on the Restaurant Reviews dataset, where short documents make smoothing more influential. For larger dataset $\alpha = 1$ is usually sufficient as they have a larger vocabulary.

For all the datasets the train and test split was 80/20 with stratification. Stratifying on the class label maintains class proportions both in the training and in test sets. The raw text was split into train/test then vectorizers were fit on the training text. This prevents leakage of test-specific vocabulary and unrealistic performance.

### 3.5. Experiment Overview

The purpose of this study's experiment is to understand what influences the Naïve Bayes classifiers' performance positively or negatively, across different datasets. All three datasets used in this study were obtained from public repositories (Kaggle) and processed using a common procedure pipeline. The experiments were implemented in Python using Google Colab, using NLTK for text preprocessing, scikit-learn for vectorization and model training and matplotlib for visualization. For each dataset the same preprocessing steps were applied. After training, each classifier was evaluated using evaluation metrics.

## 4. Results and Analysis

### 4.1. Evaluation metrics

To evaluate text classification models it is important to use metrics that capture different aspects of predictive performance. Accuracy measures overall correctness, however it often fails to reveal performance disparities between classes, like in the case of an imbalanced multi-class as the IMDb Movie dataset. Therefore, this study includes multiple complementary metrics: accuracy, precision, F1-score (macro and weighted) and confusion matrixes.

Accuracy (Figure 6) is the fraction of correctly classified instances across the entire test set, where TP is true positives, TN is true negatives, FP is false positives and FN is false negatives. This metric is informative for balanced datasets, but it can be misleading because it does not provide information about how well the model handles minority classes [Pow22]. That becomes an issue for the IMDb dataset.

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

**Figure 6.** *Accuracy formula*

Precision (Figure 7) measures how reliable the positive predictions of a classifier are. A high number shows that the model rarely misclassifies negative samples as positive. For the multi-class tasks such as News Topic and IMDb Movies datasets, it is computed per class, aggregated using macro and weighted averaging.

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

**Figure 7.** *Precision formula*

Recall (Figure 8) shows the proportion of true positive cases that are predicted correctly. This metric indicates the model's ability to pick up relevant information [Pow22]. For the IMDb Movie dataset a strong recall would show that the model can identify minority classes such as fear or disgust.

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

**Figure 8.** *Recall formula*

The F1-score metric is the harmonic mean between precision and recall. It is relevant in situations when both false positives and false negatives are important. Macro f1 treats every class equally, important for imbalanced datasets. It can uncover issues that a model could have in classifying minority classes. Weighted F1 accounts for class imbalance, so it does not penalize for bad results in minority classes. It can be useful if the overall performance is important.

For the multi-class datasets in this study confusion matrices are informative to be able to analyze error patterns. They show which classes are confused with each other, if the model is biased towards frequent classes. Confusion matrices underline how the model fails, not just how often it fails.

### 4.2. Restaurant reviews dataset results

The Restaurant Reviews dataset represents the simplest scenario in this study, having short, binary labeled texts. The average review has a length of eight words and the sentiment terms are often explicit, therefor it provides an ideal situation to examine the baseline behavior of Naïve Bayes, under different configurations.

All experiments were conducted using the same preprocessing steps and on the same training data. The goal is to identify how each configuration affects the performance and which one is the best choice for this task. Having a smaller vocabulary, the method used to extract the features could have interesting outcomes. The results include both quantitative measures and qualitative analyses to provide a clear understanding of why certain models perform better.

#### 4.2.1 Baseline Model: TF-IDF Unigrams + Multinomial Naïve Bayes

The first configuration tested for the Restaurant Reviews dataset uses MultinomialNB with TF-IDF unigram features. For the smoothing parameter Laplace was chosen. The vocabulary was restricted to 1500 TF-IDF features as it should be a suitable size for the simple and short reviews. The model achieved the following performance: Accuracy 0.9127, Precision: 0.9103, Recall: 0.9227, F1-score: 0.9165. This performance is also highly balanced on both classes, showing that the model does not favor either positive or negative reviews. These results show that the dataset is easy to classify because it has clear lexical clues. Unigrams work here because the short review usually contain isolated adjectives. In addition, TF-IDF down weights irrelevant frequent terms like “food” or “place”, as they do not carry a sentiment.

The confusion matrix in Figure 9 illustrates how the model’s error are distributed across the two classes. Most samples are correctly classified: 184 true negatives and 203 true positives. The negative reviews that were classified as positive could indicate that expressions like “not bad” were not picked up because of the unigram.

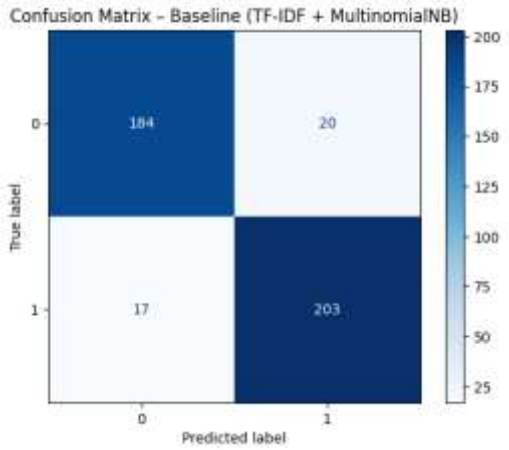



**Figure 9.** *Confusion Matrix Baseline Restaurant Reviews*

#### 4.2.2. Comparison Between all Experimental Configurations (Figure 10)

This dataset allowed to analyze different Naïve Bayes configurations through different vectorization strategies (Count vs. TF-IDF), the n-gram range and the smoothing parameter. Across all experiments the performance difference is small. This happens because the vocabulary is limited, the dataset is small and binary classification is usually an easy task.

The insignificant difference in performance between the two vectorizers show that frequency-based weighting provides no meaningful gain. So for short sentiment text, both of them perform the same. Using bigrams decrease performance introducing noise rather than useful semantic information. Smoothing controls the prior probability of unseen words. With a too low smoothing the model becomes overly confident and sensitive to rare words. A higher one makes the model to lose ability to differentiate sentiment. In this situation 1 is optimal. BernoulliNB performs well but remains inferior to MultinomialNB.

| Vectorizer | N-grams | Model | α | Max Features | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|---|---|
| TF-IDF | 1-1 | MultinomialNB | 1.0 | 1500 | 0.9127 | 0.9103 | 0.9227 | 0.9165 |
| Count | 1-1 | MultinomialNB | 1.0 | 1500 | 0.9127 | 0.9140 | 0.9182 | 0.9161 |
| Count | 1-1 | BernoulliNB | 1.0 | 1500 | 0.9104 | 0.9550 | 0.8682 | 0.9095 |
| TF-IDF | 1-1 | MultinomialNB | 0.1 | 1500 | 0.9033 | 0.9050 | 0.9091 | 0.9070 |
| TF-IDF | 1-2 | MultinomialNB | 1.0 | 5000 | 0.9009 | 0.8938 | 0.9182 | 0.9058 |
| TF-IDF | 1-1 | MultinomialNB | 5.0 | 1500 | 0.8939 | 0.8692 | 0.9364 | 0.9015 |

**Figure 10.** *Comparison Experimental Configurations*

In conclusion, all testes Naïve Bayes variants achieve great performance on this dataset, confirming that unigram models and simple bag-of-words features are sufficient for short and polarized sentiment texts.

### 4.3. News Topic Dataset Results

#### 4.3.1. CountVectorizer (unigrams) + MultinomialNB

The first experiment used a simple bag of words representation with unigrams and 5000 word features. This variant align with common classic setups for topic classification, where proper nouns, domain specific terms carry strong discriminative power. The model achieved: accuracy 0.898, macro F1 0.8976, weighted F1 0.8976. All four topics were classified with a balanced performance, however Business and Sci/Tech were slightly lower because of overlapping terminology.

The confusion matrix in Figure 11 presents a strong diagonal, indicating great predictions for the majority of instances. However, it shows an obvious misclassification between Business and Sci/Tech classes. This confusion is caused by the common terminology used by both topics.

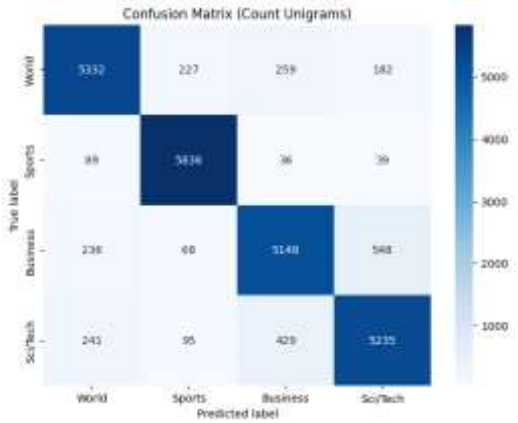



**Figure 11.** *Confusion Matrix Count Unigrams News Topic*

#### 4.3.2 TF-IDF Unigrams + MultinomialNB

The second configuration TF-IDF vectorizer, but it the same unigram representation. Results improved across all metrics: accuracy 0.908, macro F1 0.9078, weighted F1 0.9078. The slight increase reflects the benefits of down-weighting common words that appear across all news domains and create noise for CountVectorizer. TF-IDF helps emphasize terms that are specific for a domain, a useful approach for topic categorization.

### 4.3.3 TF-IDF Unigrams + Bigrams

The final variant expanded the feature space to include both unigrams and bigrams, allowing the model to capture expressions such as “stock market”. The model used 20000 features and achieved the best performance: accuracy 0.9101, macro F1 0.9098, weighted F1 0.9098. Even though the improvement is modest, it is consistent across all four classes. The bigrams can add value to the model, but because the headlines are relatively short the advantages are also limited.

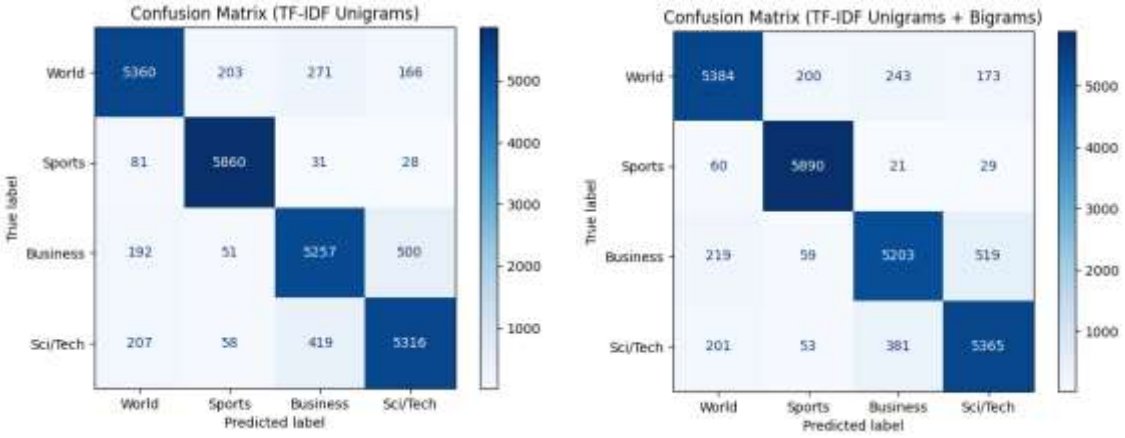



**Figure 12.** *Confusion Matrix TD-IDF unigrams vs unigrams + bigrams*

In figure 12 the two models using TF-IDF are compared using a confusion matrix. The results are similar, both showing a better distinction between classes than the countVectorizer model.

### 4.4. IMDb Movie Reviews dataset results

IMDb Movie Reviews is the most challenging dataset in this study. It is a multi-class emotion task based on short descriptions, genres with a high class imbalance. To first test the dataset only the descriptions were taken into consideration. After that, two variants were tested for classification based on both descriptions and genres.

#### 4.4.1. Baseline model

The model was trained only on descriptions, using CountVectorizer, unigrams and 5000 word features. Figure 13 presents the confusion matrix for this model. As expected for this imbalanced dataset, the model perform unevenly across emotions. The minority classes show misclassification. The anticipation class perform well, however it is often confused with sentiments like sadness or joy. This matrix illustrates that classes with more specific lexical cues like sadness or joy achieve a stronger recall.

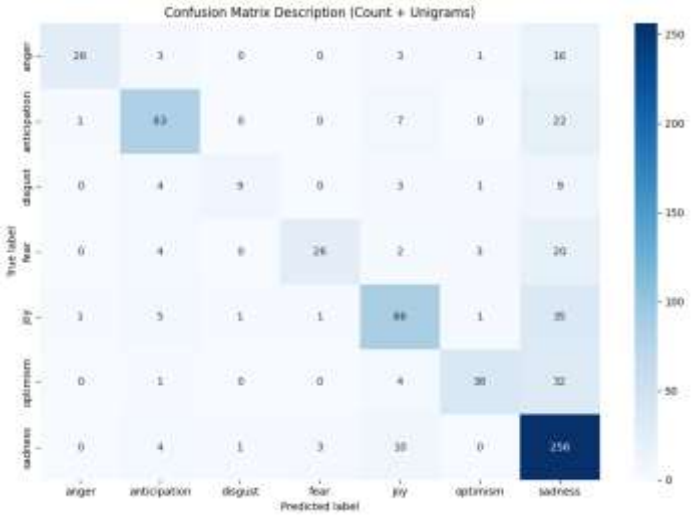



**Figure 13.** *Movies Description Confusion Matrix*

### 4.4.2. Descriptions and genres (CountVectorizer + unigrams)

This experiment investigates if adding more data like the column “genres” can improve Naïve Bayes performance for emotion classification. Genres can include more information about the emotional tone of a movie, providing informative tokens for the model. This model achieved an accuracy of 0.7161 and macro F1 of 0.6455. Certain classes benefit from this change as seen in figure 14, while the minority classes are still challenging.

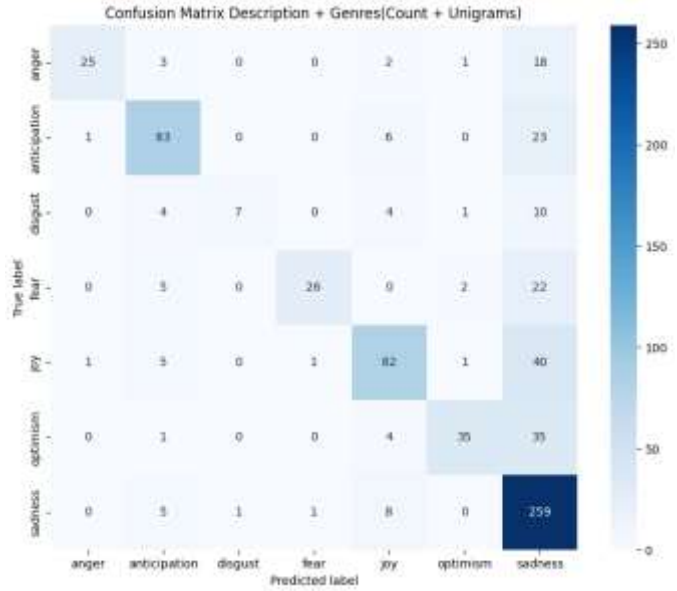



**Figure 13.** *Movies Description and Genres Confusion Matrix*

### 4.4.3. Descriptions and genres (TF-IDF + bigrams)

This configuration had the weakest performance with a 0.5549 macro recall and 0.6178 macro F1. The TF-IDF downweights rare but emotionally expressive words and because the dataset is imbalance this harms the minority emotions. The bigrams increase sparsity, as the movie descriptions are short, with an average of 27 words. In this situation, MultinomialNB does not perform well with the small continuous weights produced by TF-IDF.

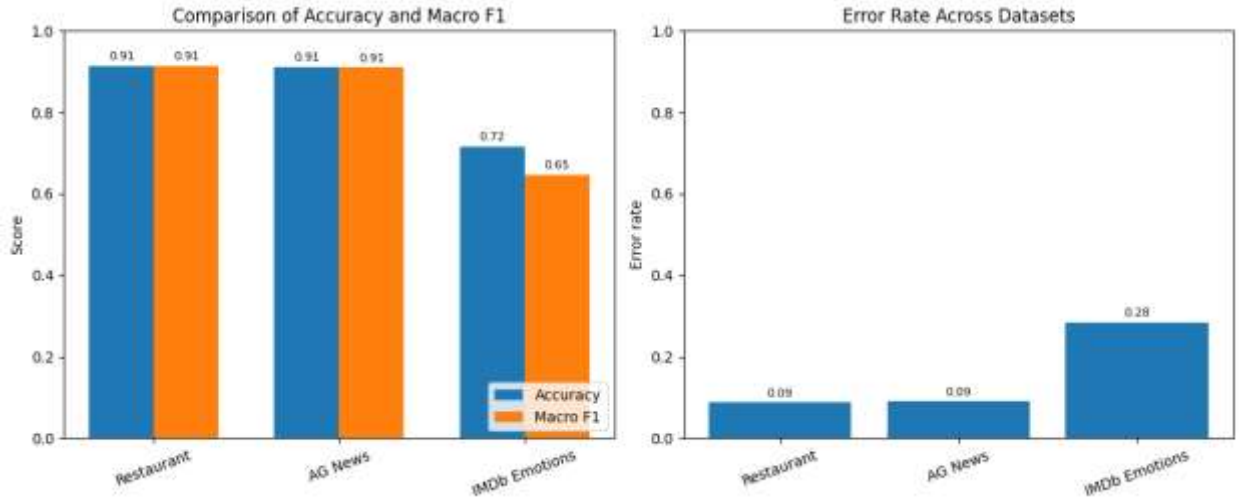
### 4.4. Comparing the results across datasets

For a better understanding of how Naïve Bayes works across different natural language processing tasks, from each dataset the best configuration was selected and compared using accuracy, macro F1 and error rates as seen in figure 14.

The Restaurant Reviews model achieved the highest performance. It is a clear reflection of the simplicity of binary polarity detection and the limited vocabulary of short review texts. The News Topic dataset still performs well despite being a four class problem. This shows that Multinomial Naïve Bayes benefits from highly distinctive lexical cues. In contrast, the IMDb movies dataset has lower results. The drop in performance is caused not only by the imbalance of classes but also because certain words are semantically close and by removing their context emotions are harder to identify.

The experiments demonstrate that naïve Bayes is effective for tasks with strong lexical separability, but the independence assumptions become limiting in nuanced affective classification.

**Figure 15.** *Comparison of result from all datasets*




## 5. Conclusions

This study evaluated the performance of Naïve Bayes classifiers across three different scenarios of text classification: short sentiment analysis (Restaurant Reviews), topic categorization (AG News) and a multi-class emotion dataset (IMDb Movies). The goal was not only to measure performance, but to also understand when and why Naïve Bayes succeeds or fails. Despite the fact that Naïve Bayes is a simple probabilistic algorithm that assumes conditional independence between words, the results prove its well-known strengths such as efficiency [Man08]. A consistent preprocessing pipeline and multiple vectorization strategies were applied to provide a better comparison between all scenarios.

Across the first two datasets, Naïve Bayes illustrates a stable and strong performance. The Restaurant Reviews dataset had an accuracy of 91%, as short texts containing clear sentiment cues are a great match for this model. Similar performance was achieved with the News Topic dataset, confirming that the model benefits from frequent and domain specific terms.

The limitations of the model were revealed by the last dataset. Multi-class emotion classification on short descriptions proved to be more difficult. The class imbalance, the contextual clues and subtle differences between the emotions vocabularies created a challenge for Naïve Bayes. These results show that the model can struggle when tasks require a deeper semantic interpretation.

All experiments underline the importance of preprocessing choices or feature representations. Count-based models were better suited for highly imbalance tasks in comparison to TF-IDF.

In conclusion, the experiments indicate that Naïve Bayes remains a reliable and efficient baseline for text classification, particularly in scenarios dominated by lexical regularities, despite the fact that its performance is influenced negatively when linguistic nuance or context dependence is present.

**Acknowledgement:**
This work is the result of my own activity, and I confirm I have neither given, nor received unauthorized assistance for this work.
I declare that I did not use generative AI or automated tools in the creation of content or drafting of this document.

## REFERENCES

[AGC] G. M. Di Nunzio, AG's Corpus of News Articles, University of Padua, 2014.
[Pow22] D. Powers, Evaluation: From Precision, Recall and F-Measure to ROC, Informedness & Markedness, Journal of Machine Learning Technologies, 2020.
[Ras14] S. Raschka, Naïve Bayes and Text Classification I – Introduction and Theory, 2014.
[MS99] C. D. Manning and H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, 1999.
[JM23] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed., Draft version, 2023.
[Wol14] J. Wolfson et al., A Naive Bayes Machine Learning Approach to Risk Prediction Using Censored Time-to-Event Data, Technical Report, University of Minnesota, 2014.
[Man08] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.