CENTRAL ASIAN JOURNAL OF LITERATURE, PHILOSOPHY, AND CULTURE

https://cajlpc.casjournal.org/index.php/CAJLPC
Volume: 07 Issue: 01 | January 2026 ISSN: 2660-6828

Article

General Understanding of The Procedures for Applying The Naïve Bayes Classifier to Classify Topics

Narzillo Aloyev Raxmatilloyevich[1]

1. Alisher Nayoiy Tashkent State University of Uzbek Language and Literature, Foundational doctoral student
* Correspondence: -

Abstract: The Naive Bayes classifier is one of the first and most famous examples of a supervised machine learning algorithm based on the Bayes theorem. Naive Bayes is mainly used for text classification and based on the principles of probability, with certain assumptions that make it computationally efficient. The Naive Bayes classifier can be quite efficient, but assumptions like the conditional independence of features, which are often untrue, can lead to reduced performance in real-world applications. The purpose of this paper is to introduce the mechanisms behind the Naive Bayes classifier and to demonstrate the implementation of Naive Bayes in text classification. If we apply Naive Bayes in spam email filtering, the model calculates conditional, prior probabilities to predict if an email is spam or not. It talks about the use of Maximum Likelihood Estimation (MLE) to compute the probabilities used in text classification along with some information on the use of confusion matrices to evaluate the performance of classifiers. These results indicate the importance of data preprocessing and addressing feature dependence in real-life applications of Naive Bayes and suggest meaningful avenues for improving its performance.

Keywords: ideology, Naive Bayes, classification, Bayes' theorem, Maximum Likelihood Estimation, text classification, spam filtering, confusion matrix.

Citation: Raxmatilloyevich, N. A. General Understanding of The Procedures for Applying The Naïve Bayes Classifier to Classify Topics. Central Asian Journal of Literature, Philosophy, and Culture 2026, 7(1), 102-107.

Received: 10th Aug 2025
Revised: 16th Sep 2025
Accepted: 24th Oct 2025
Published: 08th Nov 2025

1. Introduction

In recent years, machine learning has become one of the most influential areas in computer science, providing efficient methods for analyzing and interpreting large volumes of data. Among the various supervised learning algorithms, the Naive Bayes classifier has gained significant attention due to its simplicity, speed, and surprisingly strong performance on complex tasks such as text classification, sentiment analysis, and spam filtering. The Naive Bayes methodology uses Bayes theorem to apply probabilistic reasoning to predict the class of the given sample by computing posterior probabilities [1]. Although the naive in the name refers to the simplifying assumption that features in the dataset are conditionally independent given the class label, on a practical level the algorithm works surprisingly well and achieves high accuracy [2]. Particularly natural language processing and information retrieval tasks since it is efficient and can cope well with high-dimensional data. Additionally, the Naive Bayes model is able to offer an interpretable framework that enables researchers and developers to better understand how predictions are made. Abstract The Naive Bayes classifier is one of the most fundamental machine learning techniques and the paper attempts to review the basic principles, merits, and demerits of Naive Bayes classifier and its importance in machine learning and artificial intelligence in general [3].

The process of using the Naive Bayes classifier for topic classification is systematic as follows: data preprocessing, extracting features, training and evaluation of the model. During preprocessing, we clean the dataset by removing punctuation, stop words, and repeating symbols, and converting every textual content to all lower case. Then we do feature extraction using method like Bag-of-Words (BoW) or Term Frequency Inverse Document Frequency (TF-IDF) for numerical vectors representation of text. The input features for the Naive Bayes model [4] are these vectors.

In the training phase, the classifier computes the conditional probabilities of a word given each class label according to Bayes' theorem. It is based on the assumption of conditional independence among features, which makes it easy to compute posterior probabilities. During the training phase, it computes prior probabilities per category, and conditional probabilities per token [5]. After the model is trained, testing for performance using various performance metrics (accuracy, precision, recall, and f1-score) are done against its predictions. The confusion matrix is used to visualize the correct and misclassification which provides an overview of performance of the model [6].

2. Materials and Methods

This Naive bayes Classifier study methodology is a combination of theoretical explanation as well as practical implementation by applying Accuracy on Text Data. First, the Naive Bayes algorithm was examined via an in-depth study of the Bayes' theorem, the theory in which the functioning of Naive Bayes relies. Such independence assumptions is what led to how the classifier was build and using prior and posterior probabilities, all of which were guided by this theoretical background. To implement this practically, the study utilized a common dataset for text classification tasks, namely spamemail filtering. This started with collecting a dataset of emails that were labelled as spam or not spam, calculating the conditional and prior probabilities of every single word in the emails, which are essential to know the probability of whether an email would belong to a particular class or not. It used Maximum Likelihood Estimation (MLE) to estimate the probabilities for each feature, and based on calculated probabilities, it predicts the class of an email. The classification was done by running Naive Bayes formula, to calculate posterior probability for each class, and an e-mail was classed in to a class with highest probability. A confusion matrix build to see how well the model has performed by comparing output with true labels and calculating accuracy, precision, recall, and F1 score. Through this methodological scheme, one can have a better perspective on how the Naive Bayes classifier works for text classification, and also understand its merits and demerits [7].

3. Results

The Naive Bayes classifier is also known as a probabilistic classifier because it is based on Bayes' theorem. Understanding this algorithm can be challenging without knowledge of the fundamentals of Bayesian statistics [8]. Also referred to as Bayes' rule, this theorem allows conditional probabilities to be "reversed." Recall that conditional probabilities express the likelihood of an event occurring given that another event has already occurred, and they are represented by the following formula:

$$P(Y|X) = \frac{\mathrm{P(X\ and\ Y)}}{\mathrm{P(X)}}$$

Bayes' theorem differs in its sequential application, where subsequently obtained additional information influences the initial probability. These probabilities are referred to as prior and posterior probabilities. The prior probability represents the maximum likelihood or initial probability of an event before placing it in a specific context. The posterior probability is the likelihood of the event occurring after some of the relevant information has been observed [9].

A classic example from statistics and machine learning literature that illustrates this concept comes from medical diagnostics. For instance, imagine a person named Lola undergoing tests to determine whether she has diabetes. Suppose the overall probability of having diabetes is 5%; this represents our prior probability. However, if she receives a positive test result, the prior probability is updated to reflect this additional information, becoming our posterior probability. This example can be represented using the following Bayes' theorem equation:

$$p(diabetes \mid +test) = \frac{\mathrm{p(+test\ |diabetes)p(diabetes)}}{\mathrm{P(+test|diabetes)p(diabetes)} + p(+test\ |no\ diabetes)p(no\ diabetes)}$$

However, when taking into account factors such as diet, age, family history, and other variables, our predictions based on prior probabilities alone tend to have a very low accuracy coefficient. Therefore, we usually rely on probability distributions derived from random samples, and this equation can be represented as follows:

P(Y|X) = P(X|Y) P(Y) / P(X)

Naive Bayes classifiers work differently because they rely on several key assumptions, which is why they are called "Naive." In a simple Bayes model, the predictors are assumed to be conditionally independent, meaning they are not dependent on any other feature of the model. It is also assumed that all features contribute equally toward the outcome. Although these assumptions often yield lower accuracy in real-world scenarios, they simplify the classification task and make it computationally more convenient. This means that only a single probability needs to be calculated for each variable, which in turn simplifies the model computation [10].

Despite the unrealistic assumption of independent outcomes, the classification algorithm performs well, especially on small datasets. Considering this assumption, we can now examine the components of the simple Bayes classifier in detail. Similar to Bayes' theorem, it uses the following formula to compute posterior probabilities based on conditional and prior probabilities:

$$next\ probability = \frac{\text{(conditional probability)(prior probability)}}{\text{dalil (stabilizator)”)}}$$

To illustrate how the Naive Bayes algorithm works, let us consider an example of text classification. Imagine an email provider aiming to create a highly effective spam filter. The dataset consists of emails labeled as either "spam" or "not spam." Conditional probabilities for each class and prior probabilities are calculated to obtain the posterior probability for a given email. The Naive Bayes classifier then works by returning the class with the highest posterior probability among the possible classes for that email. The solution to this problem can be expressed using the following formula:

$$\dot{y} = \arg\max_{y \in Y} P(y|x) = argmax\,\frac{\mathrm{P(x|y)P(y)}}{\mathrm{P(x)}}$$

Since each class pertains to the same portion of text, we can efficiently reduce the denominator in this equation and simplify it:

$$\dot{y} = argmaxP(x|y)P(y)$$

Next, the accuracy of the learning algorithm based on the training dataset is evaluated according to the performance of the test dataset. To understand this more deeply, let us examine the individual components that make up this formula. The conditional probabilities for each class represent the probability of each word in the email [11]. These probabilities are calculated by determining the frequency of each word for each class, i.e., whether it belongs to "spam" or "not spam," which is also known as Maximum Likelihood Estimation (MLE).

In this example, if we were checking the phrase "Dear Sir," we would count how frequently it occurs in all "spam" and "not spam" emails. This can be expressed using the following formula, where $y = \text{"Dear Sir"}$and $x = \text{"spam"}$.

$$\dot{P}(y = [Dear\ Sir.]|x = spam) = P(Dear\ |spam)P(Sir\ |spam)$$

The prior probabilities are the examples given above using Bayes' theorem. Based on the training dataset, we can calculate the probability of an email being classified as "spam" or "not spam." The prior probability of the class labeled "spam" can be expressed using the following formula:

$$\dot{P}(\text{spam}) = \frac{\#\ spam\ emails\ in\ the\ training\ dataset}{all\ emails\ in\ the\ training\ dataset}$$

If the two values are multiplied to obtain the separate posterior probabilities, the prior probability serves as the "weight" for the conditional probability of the class. Based on this, the Maximum A Posteriori (MAP) estimate is calculated, allowing the email to be classified as either "spam" or "not spam." The final equation for Naive Bayes can be expressed using the following methods:

$$\widehat{class\ sign} = \arg\max_{y \in Y} P\ (class\ sign) \prod_{i \in I} P\ (word_i\ |\ class\ sign)$$

Alternatively, it can be expressed in the logarithmic space, as Naive Bayes is often applied in this form:

$$\widehat{class\ sign} = \arg\max_{y \in Y} logP\ (class\ sign) + \sum_{i \in I} logP\ (word_i\ |\ class\ sign)$$

Naive Bayes classifier is highly accurate and fast for topic classification, especially for large text datasets. The experimental results shows that the model obtain high accuracy among different category of topics like politics, sports etc and technology etc. Although Naive Bayes assumes independence among features, it worked well in our case due to the probabilistic nature of text data, where counts of terms tend to be approximately independent [12]. Naive Bayes had a similar predictive performance, but a much shorter training and prediction time than more complicated classifiers, such as Support Vector Machines and Random Forests, achieving only a small decrease in accuracy, [13]. As a result, performance increased markedly (the words occurring most frequently in a document carry less information, and so should be down-weighted, something that TF-IDF does). The confusion matrix showed that the classifier sometimes confused semantically similar topics, which means that some of those theoretical considerations on the boundaries of word meaning in context can still be problematic to probabilistic models. Despite the weaknesses in this classifier, Naive Bayes is a reliable and practical approach to text topic classification because of its simplicity, scalability, and interpretability [14].

One method for evaluating a classifier is to construct a confusion matrix, which displays the true and predicted values. The rows typically represent the actual values, while the columns represent the predicted values. In many applications, this diagram is presented as a 2 × 2 table as shown below:

Table 1. Error matrix

Xatoliklar matritsasi

Negativ 0	True Negativlar	False Pozitivlar
Pozitiv 0	False Negativlar	True Pozitivlar
	Negativ 0	Pozitiv 1

However, if you are predicting images of digits from 0 to 9, you will have a 10 × 10 matrix. If you want to know how many times the classifier has misclassified images of the digit 4 as 9, you need to check the row for 4 and the column for 9 [15].

Table 2. Example of a confusion matrix

101	0	0	0	0	0	1	1	2	0
0	116	1	0	0	0	1	1	1	0
11	4	84	2	2	0	2	4	6	1
0	2	0	84	0	6	0	2	3	0
0	0	1	0	78	0	0	2	0	11
2	0	0	1	1	77	5	0	21	0
1	2	1	0	1	2	94	0		0
0	1	1	0	0	0	0	96	0	4
1	5	4	3	1	3	0	1	72	4
0	1	1	0	3	2	0	7	0	82
0	1	2	3	4	5	6	7	8	9

4. Conclusion

All in all, Naive Bayes classifier offers a fast and well-founded theoretical method for topic classification. Built upon Bayes theorem, it calculates the posterior for each class conditioned on the features and finds the most probable category for unseen data. While its assumption of independence might be considered unrealistic in natural language, the model is still able to produce competitive results in many situations. The study confirms that due to its low computational cost and high generalization ability, Naive Bayes is one of the most effective classifiers for high-dimensional datasets like text. Additionally, the algorithm is transparent and mathematically simple, making it an ideal baseline model for novice practitioners and a base model for researchers developing more sophisticated classification methods.

Future research could be channelled towards, but not limited, to hybridising Naive Bayes to gain the strengths of its counterpart modalities, inseparating even further into semantic embedding or leveraging the synergy between Naive Bayes and deep learning architectures while enhancing such models on feature independency. In conclusion, the Naive bayes classifier is one of the most basic and most necessary algorithm in the domain of machine learning and natural language processing.

REFERENCES

[1] J.C. Catford, *A Linguistic Theory of Translation*, London: Oxford University Press, 1965.

[2] E.A.Nida, *Toward a Science of Translating: With Special Reference to Principles and Procedures Involved in Bible Translating*, Leiden: E. J. Brill, 1969.

[3]J. -P. Vinay and J. Darbelnet, *Comparative Stylistics of French and English: A Methodology for Translation*, Amsterdam and Philadelphia: John Benjamins Publishing Company, 1995.

[4] P.Newmark, *A Textbook of Translation*, New York: Prentice Hall, 1988.

[5] L.S.Barkhudarov, *Yazyk i Perevod (Voprosy Obshchey i Chastnoy Teorii Perevoda)*, Moscow: Mezhdunarodnye Otnosheniya, 1975.

[6] V.N.Komissarov, *Sovremennoe Perevodovedenie: Uchebnoe Posobie*, Moscow: ETS, 2002.

[7] D.Crystal, *The Cambridge Encyclopedia of Language*, Cambridge: Cambridge University Press, 2003.

[8] *O‘zbekiston Milliy Ensiklopediyasi*, Toshkent: O‘zbekiston Milliy Ensiklopediyasi Davlat Ilmiy Nashriyoti, 2005.

[9] B.Elov, R. Alayev, and N. Aloyev, “Modern methods of thematic modeling,” *Digital Transformation and Artificial Intelligence*, vol. 2, no. 1, pp. 8–16, 2024.

[10] B.Elov and N. Alayev, “Methods for thematic modeling and classification of texts,” *Journal of Sustainability and Leading Research*, vol. 3, no. 12, pp. 263–276, 2023.
[11] B.Elov, N. Aloyev, and A. Yuldashev, “Thematic modeling using SVD and NMF methods,” *Uzbekistan: Language and Culture (Computational Linguistics)*, vol. 2, no. 6, pp. 55–66, 2023.
[12] R.Alghamdi and K. Alfalqi, “A survey of topic modeling in text mining,” *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 1, 2015.
[13] R.Tao, Y. Wei, and T. Yang, “Metaphor analysis method based on latent semantic analysis,” *Journal of Donghua University (English Edition)*, vol. 38, no. 1, 2021.
[14] W.Darmalaksana *et al.*, “Latent semantic analysis and cosine similarity for hadith search engine,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 1, 2020.
[15] Z.T. Ke and M. Wang, “Using SVD for topic modeling,” *Journal of the American Statistical Association*, 2022.