Spam Detection with Naive Bayes

*Note: Sub-titles are not captured for https://ieeexplore.ieee.org and should not be used

1st MD Sayem
dept. of Electronic Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
md.sayem@stud.hshl.de

***Abstract*—Spam detection is an important task in the current digital age, especially in the areas of natural language processing and privacy security, as unwanted messages can disrupt digital communication.This research uses a machine learning-based method to classify SMS messages as either SPAM or HAM (legitimate messages). A data set called SMS Spam Collection will be used, which includes over 5,500 labeled messages .Naive Bayes a machine learning algorithm will be used to detect spam messages. Naive Bayes is renowned for its simplicity, fast training capabilities and text classification performance. Through data pre-processing, feature extraction and possibility based modeling, the Naive Bayes will be trained and learn to effectively distinguish between Spam and non-spam (HAM) messages.The test results show that the model achieves high accuracy and precision with unseen data, proving the suitability of the Naive Bayes algorithm for spam detection.**

***Index Terms*—component, formatting, style, styling, insert.**

I. Introduction

In the current era, email communication has established itself as a fast, affordable and effective medium that has become important for everyone from individual users to corporate organizations.While the popularity of email in information exchange has provided benefits, it has also created some negative aspects.One of these problems is spam email, which sends unnecessary and confusing messages to users' inboxes and disrupts their work pace and focus. [1]

Research has shown that an average user receives 40 to 50 emails per day, of which 60 to 70 percent may be spam.In this situation, it becomes difficult to manually filter spam.Many times these spam emails are used by hackers to hack someone's account, spread malware, or create unnecessary traffic on the network [1].To address this challenge, automated and intelligent solutions are needed, for which machine learning-based algorithms are an effective way.

Previously, blocking messages from certain email IDs or domains was the solution, but now those strategies have become ineffective.Because, spammers are continuously changing their strategies and techniques to send spam messages and act legitimate.Machine learning-based methods have proven particularly effective in solving this problem. [2]

Among machine learning algorithms, Naive Bayes classifier is a popular and effective method for spam detection.It works on the basis of probability and can be trained very quickly, which makes it provide effective results on larger datasets [1] [2].

In this paper, we demonstrate an efficient model for spam recognition using the Naive Bayes classifier.A standard data set is used where email messages are tagged as either Spam or Ham.After preprocessing, feature identification and training, the model's performance is evaluated through accuracy, precision,recall and F-score.

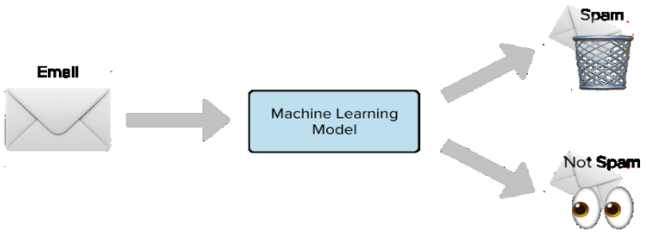

Fig. 1. Classification into Spam and non-spam [2]

This image in fig:1 illustrates the process of a simple spam recognition.An input email is first analyzed by a machine learning model. The model determines whether the message is spam or Ham based on the trained data.If the email is marked as spam, it is sent directly to the spam folder.On the other hand, if the email is not a spam, then it is considered as ham or valid and important and is brought to the user's attention.

II. Background and Related Work

Email spam is a long-standing problem that wastes time and sometimes resources by sending users annoying messages and unwanted information through their email systems.Initially, spam recognition mainly used by fixed rules and blacklists, but these techniques can be easily avoided by spammers [1].

Naive Bayes algorithm has been used for a long time in email spam filtering.Harisinghaney et al. (2014) conducted a comparative analysis of Naïve Bayes, KNN, and Reverse DBSCAN algorithms and showed that, even without preprocessing, Naïve Bayes provides significant accuracy 87 percent.Similarly, Mohamad and Selamat (2015) applied a hybrid method of TF-IDF and Rough Set Theory in their study

Identify applicable funding agency here. If none, delete this.

and obtained good results.This method has proven to be very effective, especially in feature selection. [3] [4]

Apart from Naive Bayes, various studies have attempted to improve spam recognition using techniques such as Support Vector Machine (SVM) Extreme Learning Machine (ELM) Decision Tree and Artificial Neural Network.For example, Feng et al. (2016) proposed a hybrid SVM-NB model, which was tested on the Chinese spam email dataset and found that it performed better than either single SVM or NB. [5]

In addition, Naive Bayes has been used in some studies in combination with Particle Swarm Optimization (PSO) for better optimal results.Idris et al. (2015), in their research, improved the accuracy of spam classification up to 83.20 percent by combining PSO with the negative selection algorithm. [6]

III. Machine Learning

Machine learning is a cutting-edge technology of our modern world of time that gives computer systems or a machine the ability to learn automatically through data analysis. Machine Learning is a branch of artificial intelligence (AI) where machines learn from experience and are able to make future decisions without following any specific rules like human. In general programming, specific instructions are given for each step. Machine learning models learn by identifying hidden patterns and making future decisions using various algorithms and statistical models such as Linear regression, decision-tree, neural networks and etc. Currently, machine learning technology is being widely used in many fields including healthcare, banking, cybersecurity, robotics, and language processing. It is considered to be one of the driving forces of our industrial revolution, which has opened up a new horizon of automation and intelligence in our lives. [7].

A. Supervised learning

Supervised learning is a type of machine learning where each input data has a specified output. The model learns based on this input-output pair and is able to predict the relevant output after receiving new input.The most common phases of supervised learning include classification, such as email spam or ham, and regression, such as predicting future sales. The effectiveness of this method depends on the quantity and quality of the training data set.It has been successfully used in healthcare, financial institutions, and e-commerce sectors. [8] [9]

B. Unsupervised learning

Unsupervised learning is a type of learning method where the input data does not have any levels or predetermined outputs. In this method, the algorithm analyzes the structural features of the data to find clustering patterns or hidden relationships.Clustering, Association Rule Learning and Dimensionality Reduction fall into this type of learning and are mainly used for data exploration, segmentation, and social network analysis. [8] [9]

C. Reinforcement learning

Reinforcement learning is a machine learning method in which an agent interacts with its environment and is rewarded or punished according to the results of its actions. The goal of this learning process is to develop policies by maximizing rewards.The method has widespread use in gaming, robotic control, and automated driving.The Reinforcement Learning helps in building models for adaptive decision making in different real environments. It is a long-term learning process where the agent analyzes feedback from its past experiences and determines future actions. [7]

IV. Dataset

The SMS Spam Collection dataset is download from kaggle.com. The dataset is a collection of 5,574 SMS messages. Each messages are represented as a single line in the file. Each line contains two fields, the first field (v1) is a label indicating whether the message is ham (legitimate) or spam, and the second field (v2) contains the raw text of the message. The dataset is provided in a plain text format, which making it easy and straight-forward to load and process for machine learning tasks. There are no missing values in the dataset which is very efficient for data pre-processing and feature extraction.

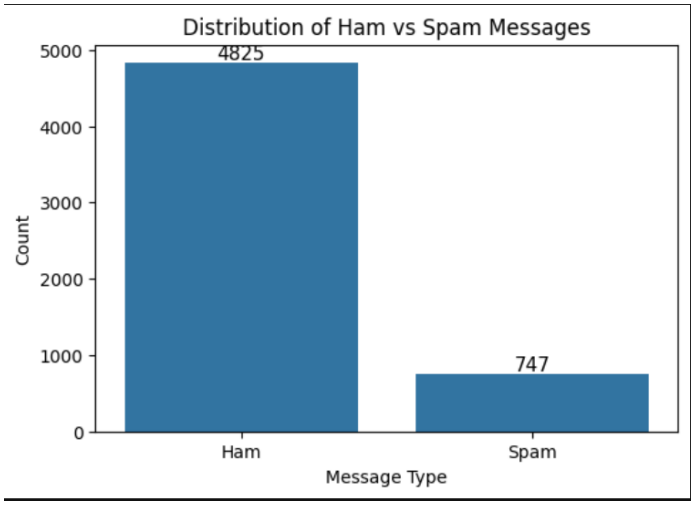

Fig. 2. Classes of The Dataset

V. Naive Bayes

The Naive Bayes algorithm is a machine learning classification method that works based on Bayes' theorem. The main feature of this algorithm is that it assumes each feature as independent, which is why it is called Naive.It is widely used in text classification, such as spam detection, because it can work quickly and efficiently using less data.Naive Bayes performance is very good in many cases, even when the features are dependent on each other. One of the biggest advantages of Naive Bayes is that it works very fast in both training and prediction, which is very useful for large datasets. It can also perform well on noisy data.Its practical applications include email filtering, document categorization,

and sentiment analysis, but it has limitations, such as the fact that the results can be less reliable when features depend heavily on each other.Additionally, continuous features require special transformations such as Gaussian Naive Bayes, yet Naive Bayes is being used as a popular classifier due to its simplicity, scalability, and excellent performance. [10] [5]

A. Types of Naive Bayes Classifiers

There are different variants of the Naive Bayes algorithm that are suitable for different types of data.

1) Multinomial Naive Bayes: Multi-Nominal Naive Bayes Classifier is the most used for SMS or email spam detection , which is very effective for determining things like word count or word density in a SMS or an email. It assumes that each feature, such as the number of words in a document, follows a multi-nominal distribution. This model can efficiently model the density of each word in a SPAM versus a normal HAM message, making it successfully used for spam filtering. Multi-nominal Naive Bayes is particularly suitable for document classification, especially in natural language processing applications such as email filtering and spam recognition. [11] [12].

2) Bernoulli Naive Bayes: Another important type is the Bernoulli Naive Bayes, which is used for binary features such as whether a word is present or absent in a message. Multi-nominal models, which look at the density of words, are associated only by considering the presence or absence of the word. The Bernoulli Naive Bayes model, where the number of times a word appears is not important, but whether the word is present or absent is sufficient. Although it is not widely used in spam filtering, it is very useful in cases where yes or no type features are used [12].

3) Gaussian Naive Bayes: The third type is Gaussian Naive Bayes, which is designed for continuous data. It assumes that each feature follows a Gaussian normal distribution and is commonly used in classification where the input values are numbers, such as temperature, age, birthday.This model is not widely used in spam detection because spam data is usually discrete, but it is very useful in cases where numerical input needs to be analyzed [13].

In this study, the Multinomial Naive Bayes model was selected from among the three Naive Bayes classifiers because it is the most effective and widely used in text-based SPAM identification. Our dataset is an SMS spam collection consisting mainly of message words, where the presence and density of different words in each message is important.Multi-nominal Naive Bayes works by taking into account the density or term frequency of the word, so it is very suitable for solving such problems. On the other hand, the Bernoulli Naive Bayes model works only based on the presence or absence of words and the Gaussian model is suitable for continuous or numerical input, which is not applicable to this project. The model is the best choice for our spam filtering system, which provides fast training, high reliability and simple implementation.

B. Mathematical Foundation and Basic Principle

Mathematically, it determines the probability using the probability of each class using Bayes' Theorem: formula shown

$$P(C \mid X) = \frac{P(X \mid C) \cdot P(C)}{P(X)}$$

Fig. 3. Naive Bayes Theory

in fig:3, where C is the class and X is the feature vector extracted from the SMS contentes. The Naive Bayes classifier assumes that the features of the SMS x1,x3,x3..... in X are independent [5].

C. Bayes' Theorem Applied to Spam Classification

$$P(\text{Spam}|\text{Clues}) = \frac{P(\text{Clues}|\text{Spam}) \times P(\text{Spam})}{P(\text{Clues})}$$

Fig. 4. Bayes Theorem Simplified for Spam Detection

The picture in fig:4 shows again Bayes' Theorem simplified for spam classification. It helps calculate the probability that an SMS is SPAM or HAM based on the clues (words) it contains.

P(Spam—Clues) : The probability of SMS is Spam, if the sms contains certain word or feature.

P(Clues—Spam) : It means how often these words show up in spam SMS. For example, how often the word "Free" exist in Spam Sms.

P(Spam) : How many SMS are Spam in total.

P(Clues) : How often do the words show up in the SMS (SPAM or HAM).

After the calculation, the model picks the class (SPAM or HAM) with the higher probability and marks the message as that type.

VI. Methodology

A. Required Libraries and Tools

This figure in fig:5 shows the Python libraries that are imported for implementing the Naive Bayes model to classify spam SMS.This Section includes libraries for data manipulation (pandas, numpy), text processing (re, string), and visualization (matplotlib, seaborn). For machine learning, scikit-learn provides modules for model training MultinomialNB, for feature extraction TfidfVectorizer, data splitting traintestsplit, cross-validation, and performance evaluation using metrics like accuracy, confusion matrix, and classification report.

B. Data Preprocessing

Before training the model, the raw text data is converted into a structured form that is suitable for machine learning.

```python
# Data manipulation and analysis
import pandas as pd
import numpy as np

# Text processing and regex
import re
import string

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Machine Learning
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

Fig. 5. Required Libraries and Tools

This step basically involves text cleaning and extracting relevant features, and converting each message into a vector of numbers.

1) Tokenization and Lowercasing: The text data is divided into tokens and all characters are converted to lowercase. This task is done automatically by scikit-learn's TfidfVectorizer tool, which automatically tokenizes the input text and converts all words to lowercase.This eliminates the confusion of the same word having multiple forms and improves the model's performance.

2) Punctuation and unwanted charracter Removal: Using TfidfVectorizer and Python's re (regular expression) library, many unnecessary or unwanted symbols and characters have been removed, such as commas, dots, slashes, special symbols, etc., whose presence does not contribute to text classification, making it possible to more accurately distinguish between spam and ham messages.

3) STOP words Removal: In addition, the data pre-process includes a stopword removal using a list of commonly used words such as I,me, my, you, are, was, the, and, or no, etc., which do not provide any useful information for identification, so they are stored in a set and filtered out during text cleaning.

C. Feature Engineering

For the proper functioning of the spam detection model, it is not enough to just clean the text data. Selecting and transforming relevant features is very important. In this project, mainly two types of features have been extracted: text-based features and mathematical features (Message length)

1) Construction of Feature Matrix: First, a Bag of Words (BoW) features is extracted from the text. Then it is stored in a variable named Xbow. This matrix is then converted into a sparse format using csrmatrix(). It can handle large datasets. After that the length of each message was calculated and also reshaped into a column vector using reshape(-1, 1). This column was also converted into a sparse format.Both the BoW vectors and message length feature were combined using the hstack() function to form a final feature matrix named Xcombined, which will be later used for model training and evaluation.

2) Vectorization Method Used: The text messages were transformed into numbers using a vectorizer (TfidfVectorizer)

```python
# Convert BoW list to sparse matrix
X_bow_matrix = csr_matrix(X_bow)

# Reshape message length and convert to sparse
length_col = df['length'].values.reshape(-1, 1)
length_sparse = csr_matrix(length_col)

# Combine BoW vectors with message length
X_combined = hstack([X_bow_matrix, length_sparse])
```

Fig. 6. Combining Text Features and Message Length for Model Input

earlier in the notebook. The output was saved in a variable called Xbow, which was later used to build the final feature matrix.

D. Model Training and Validation

After preparing the final feature matrix, the dataset was split into training and testing subsets using an 80:20 ratio. This allowed the model to be trained on a majority portion of the data while reserving a smaller portion for evaluating its generalization performance.The stratify=y parameter was used to ensure that both the training and testing sets maintain the balanced class distribution (Ham And Spam SMS).

1) Model Selection and Hyperparameter Tuning: The Multinomial Naive Bayes classifier was chosen for this task because it is efficient and effective in handling discrete features such as word counts. To find the best smoothing parameter (alpha), a grid search with 5-fold cross-validation was performed using GridSearchCV.

```python
#  Define model and parameter grid
model = MultinomialNB()
param_grid = {'alpha': [0.01, 0.1, 0.5, 1.0, 2.0]}

#  Use GridSearchCV to find best alpha
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)
```

Fig. 7. Model Selection and Hyperparameter Tuning

2) Best Model Selection and Saving: The best-performing model and corresponding alpha value were extracted from the grid search results. Both the model and the vectorizer were then saved using joblib for future use.

E. Cross-Validation Results Visualization

The graph in fig :8, describes how the model's cross-validation accuracy changes when different values of the smoothing parameter alpha changes in the Naive Bayes classifier. The blue line represents the average accuracy across five validation folds for every alpha value: 0.01, 0.1, 0.5, 1.0, and 2.0.

From the fig:8, we can see:

Accuracy was stable between alpha values 0.01 and 1.0.

The best performance was reached at alpha = 1.0, where the model reached n accuracy of 97.94

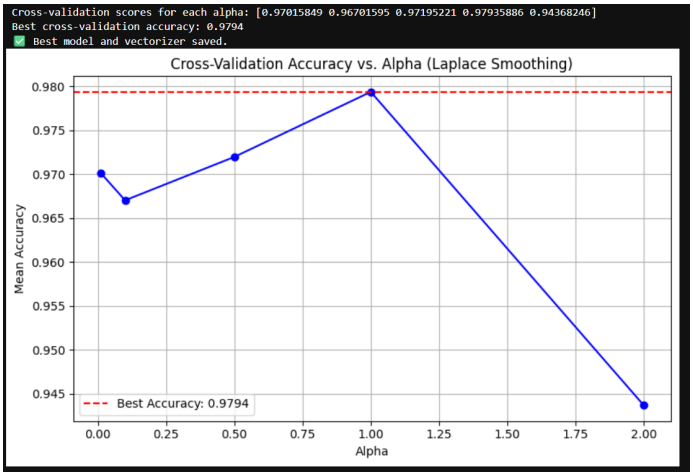

Fig. 8. Cross-Validation Accuracy vs. Alpha

At alpha = 2.0, the accuracy dropped significantly at 94 percent, showing that too much smoothing harms the model's ability to learn useful patterns.

The red dashed line indicates the best accuracy, helping visualize which alpa gained the best result.

This analysis helped in selecting the most suitable value of alpha (1.0) for choosing the final model.

F. Test Results

After selecting the best model using cross-validation, its performance was evaluated on the unseen 20 percent test data. The results demonstrate that the model generalizes well and is now capable of accurately identifying spam messages. The key performance metrics are as follows:

Metric	Value
Accuracy	0.977578
Precision	0.936620
Recall	0.892617
F1 Score	0.914089

Fig. 9. Test Result

Accuracy: 97.76 percent – it means that the model correctly predicted nearly 98 out of every 100 messages.

Precision: 93.66 percent– it means that when the model predicted a message as spam or ham , it was correct 94 percent of the time.

Recall: 89.26 percent – it implies the model successfully recognized about 89 percent of spam messages.

F1 Score: 91.41 percent – a balance between precision and recall, representing the model's overall effectiveness.

These scores implies that the spam classifier is highly accurate, with a strong balance between detecting spam and avoiding false positives.

G. Confusion Matrix For Test Data

The confusion matrix in the figure:10 shows how well the model performed on the test data by comparing the actual labels with the predicted labels.

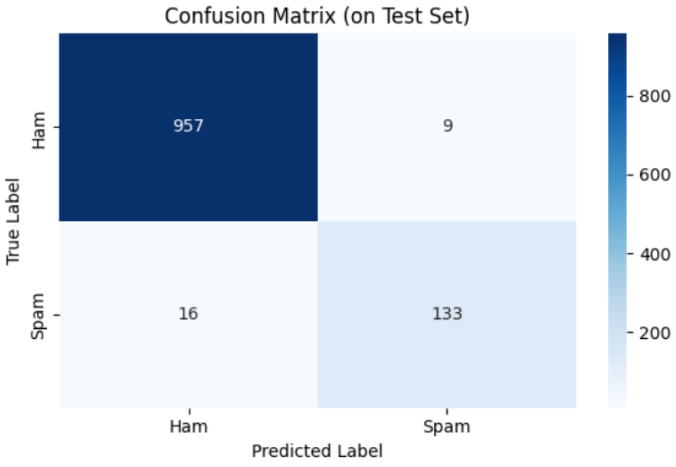

Fig. 10. Confusion Matrix For Test Data

957 messages were correctly predicted as ham .

133 messages were correctly predicted as spam.

9 ham messages were wrongly predicted as spam (false positives).

16 spam messages were wrongly predicted as ham (false negatives).

This implies that the model is very good at identifying both spam and ham messages. But like other classifiers, it made a few small mistakes. Overall, the amount of correct predictions is much higher than the incorrect ones.It is showing that the model prediction is highly accurate

VII. Conclusion

In this project, a spam detection system was developed with a machine learning based approach using the Multinomial Naive Bayes. The dataset was preprocessed by several aspects to make suitable for training like cleaning the text, removing stop words, and extracting useful features like word frequency and message length. These features were then combined to train the model.

To improve the model's performance,cross-validation is used where different values of the smoothing parameter (alpha) were tested then the best alpha value was selected, and the final model achieved high accuracy on the test data.

The results showed that the model can now effectively identify spam messages, with good precision, recall, and F1 score. The confusion matrix also confirmed that most messages were classified correctly. Overall, the model is simple and fast making it well performing for spam filtering tasks.

References

[1] K. Agarwal and T. Kumar, "Email spam detection using integrated approach of Naïve Bayes and particle swarm optimization," in Proc. 2018 2nd Int. Conf. Intell. Comput. Control Syst. (ICICCS), pp. 685–690, 2018.

[2] N. Kumar, S. Sonowal et al., "Email spam detection using machine learning algorithms," in Proc. 2020 2nd Int. Conf. Inventive Res. Comput. Appl. (ICIRCA), pp. 108–113, 2020.
[3] A. Harisinghaney, A. Dixit, S. Gupta, and A. Arora, "Text and Image Based Spam Email Classification Using KNN, Naïve Bayes and Reverse DBSCAN Algorithm," 2014 International Conference on Optimization, Reliability, and Information Technology (ICROIT), pp. 153–155, IEEE, 2014.
[4] M. Mohamad and A. Selamat, "An Evaluation on the Efficiency of Hybrid Feature Selection in Spam Email Classification," 2015 International Conference on Computer, Communications, and Control Technology (I4CT), pp. 227–231, IEEE, 2015.
[5] W. Feng, J. Sun, L. Zhang, C. Cao, and Q. Yang, "A Support Vector Machine Based Naive Bayes Algorithm for Spam Filtering," 2016 IEEE 35th International Performance Computing and Communications Conference (IPCCC), pp. 1–8, IEEE, 2016.
[6] I. Idris et al., "A Combined Negative Selection Algorithm–Particle Swarm Optimization for an Email Spam Detection System," Engineering Applications of Artificial Intelligence, vol. 39, pp. 33–44, 2015.
[7] O. Simeone, "A Very Brief Introduction to Machine Learning With Applications to Communication Systems," IEEE Transactions on Cognitive Communications and Networking, vol. 4, no. 4, pp. 648–664, Dec. 2018.
[8] S. Badillo, J. E. B. Blais, L. J. Pomares, and J. A. P. Alonso, "An Introduction to Machine Learning," Clinical Pharmacology Therapeutics, vol. 107, no. 4, pp. 871–882, Apr. 2020.
[9] M. Kumar, R. Singh, and A. K. Jaiswal, "A Conceptual Introduction of Machine Learning Algorithms," Proceedings of the 1st International Conference on Intelligent Computing and Research Trends (ICRT), IEEE, pp. 132–137, 2023; [2] S. Badillo et al., "An Introduction to Machine Learning," Clinical Pharmacology Therapeutics, vol. 107, no. 4, pp. 871–882, Apr. 2020.
[10] F.-J. Yang, "An Implementation of Naive Bayes Classifier," in Proc. 2018 Int. Conf. Comput. Sci. Comput. Intell. (CSCI), Las Vegas, NV, USA, 2018, pp. 301–306, doi: 10.1109/CSCI46756.2018.00065.
[11] I. Rish, "An empirical study of the naive Bayes classifier," in Proc. IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, Seattle, USA, vol. 3, no. 22, pp. 41–46, 2001.
[12] G. Singh, B. Kumar, L. Gaur, and A. Tyagi, "Comparison between Multinomial and Bernoulli Naïve Bayes for Text Classification," in Proc. 2019 Int. Conf. Autom., Comput. and Technol. Manag. (ICACTM), London, UK, 2019, pp. 593–596, doi: 10.1109/ICACTM.2019.8776800.
[13] H. Kamel, D. Abdulah, and J. M. Al-Tuwaijari, "Cancer Classification Using Gaussian Naive Bayes Algorithm," in Proc. 2019 Int. Eng. Conf. (IEC), Erbil, Iraq, 2019, pp. 165–170, doi: 10.1109/IEC47844.2019.8950650.