Assignment 1 : Spam Email Detection Using Naive Bayesian

Majid Rostami
Michigan Technological University
majidr@mtu.edu

Ayush Juvekar
Michigan Technological University
aajuveka@mtu.edu

Aditya Karle
Michigan Technological University
arkarle@mtu.edu

Shreyash Waghdhare
MTU
spwaghdh@mtu.edu

Abstract – This report outlines our methodology and results for a project aimed at detecting spam emails. We employed three methods. In the first method, we focused solely on the email subject text. By removing punctuation, stopping words, and tokenizing sentences, we utilized multinomial naïve Bayes and Gaussian naïve Bayes to train our model. We evaluated the model's prediction capabilities to distinguish spam emails by reporting accuracy, confusion matrix, precision, recall, and F1 score. In the second method, we balanced the label data to achieve a more uniform distribution, resulting in a slight improvement in results. In the third method, we incorporated additional features into the model beyond the subject text; however, "context" and "logical flow" did not significantly impact model accuracy.

Index Terms – spam detection, Naïve Bayesian, Gaussian Naïve Bayesian , digital fraud, scams.

INTRODUCTION

Cybersecurity is a critical aspect of digitalization, impacting all areas of life from goods purchasing to banking. Detecting spam emails is a crucial step in preventing digital fraud and scams. Machine learning has proven to be a powerful tool for this task. This study implements Naive Bayes classifiers, Multinomial NB and Gaussian NB, to distinguish spam from legitimate emails using word frequency patterns. Modern spam emails generated AI attempt to turn around keyword-based filtering by creating contextual messages that convey fraudulent content without using common keywords. Another challenge is that input data may suffer from class imbalance, which can bias the model towards a specific class; for example, if the training data contains more non-spam emails, the model may not be skeptical enough, causing spam emails to leak into users' inboxes. In this paper, we initially utilize the power of Naive Bayesian classifiers, which use statistical techniques to determine the frequency of words in spam and non-spam emails, using only text as a feature for learning. As a novelty, we use additional features based on "context detection" and "logical flow" detection, hoping these features will facilitate classification. Moreover, we attempt to perform data balancing before training to compare the results for this specific dataset.

METHODOLOGY

The imported data contains four columns: number of words in the email, label name (ham or spam), subject text and label number (0 for ham or 1 for spam), respectively, which we use the third and fourth ones.

Since data incorporate punctuation and stopping words such as “and”, “or”, “to”, “from” and so on, we perform one step on cleaning by using “nltk” library. By using CountVectorizer and fit transform, to convert text to numerical data. Then we split the input data into training and testing parts by 80% and 20%.

Our learning model is based on Bayes’ Theorem using the following equation to calculate the probability of class when we give it the features.

$$P(C|X) = \frac{P(X|C).P(C)}{P(X)}$$

$P(C)$ is probability of class before seeing the feature, $P(X)$ is probability of features, and finally $P(X|C)$ is likelihood of the features given the class. The difference between the Multinomial Naive Bayes and the Gaussian Naive Bayes classifiers is that the former uses discrete probability distributions, while the latter uses a Gaussian distribution and considers continuous features.

In addition to the base method, we explored two additional approaches. Since our dataset suffers from class imbalance (71% ham and 29% spam), in the first method, we applied data balancing using RandomOverSampler, which balances the dataset by generating synthetic samples for the minority class.

Moreover, as a preliminary exercise on how to address context-based spam emails designed to bypass common keyword-based detection methods, we explored incorporating "context detection" and "logical flow" analysis into our methodology. We included these as additional features in our model alongside preprocessed text.

To achieve this, we counted the number of words belonging to predefined categories in each email and assigned a single label based on the dominant category.

Below are some example excerpts:

contexts = **{"finance":** ["stock", "market", "downturn", "investment", "loan", "credit],
"technology": ["iphone", "innovative", "features", "software"],
"job_offers": ["job", "career", "opportunity", "hiring", "resume", "recruitment"],
"urgent_requests": ["asap", "immediate action", "respond quickly", "time-sensitive"]}

logical_flows = {
"cause_and_effect": [
"because", "since", "as a result", "therefore", "consequently"],
"problem_and_solution": [
"problem", "solution", "resolve", "address", "issue", "remedy", "solve", "fix", "answer", "approach"],
"general_to_specific": [
"for example", "for instance", "such as", "including", "to illustrate", "in particular", "namely", "specifically", "e.g.", "i.e."]}

The rationale for using this approach is that spam emails often follow specific logical patterns to manipulate recipients. They frequently use cause-and-effect, problem-and-solution, and general-to-specific structures to create a sense of urgency, introduce fake problems, and offer deceptive solutions or deals.

For instance, a spam email might state, "Due to a security breach, your account has been locked," or "Because of this exclusive deal, you can save 70% today!"—both designed to pressure the reader into acting quickly.

The problem-and-solution tactic is particularly common in scams, with messages like, "Your PayPal account has been compromised. Click here to secure it immediately," or "Struggling with hair loss? Our revolutionary treatment is the answer!" These strategies aim to trigger fear or excitement, prompting users to take immediate action without critical thinking.

In conclusion, we developed three versions of the code: a baseline model, a version with balanced input data, and one incorporating additional features.

RESULTS

We use four key metrics to evaluate our model: accuracy, precision, recall, and F1-score. Accuracy measures the overall fraction of correct predictions and does not discriminate between false positive and false negative. While it is simple, it cannot give us further details. Precision tells us how many of the predicted spam emails are truly spam, focusing on minimizing false positives. Recall, on the other hand, measures how many actual spam emails were correctly identified, focusing on minimizing false negatives. The F1-score equation uses both precision and recall, providing a more general understanding.

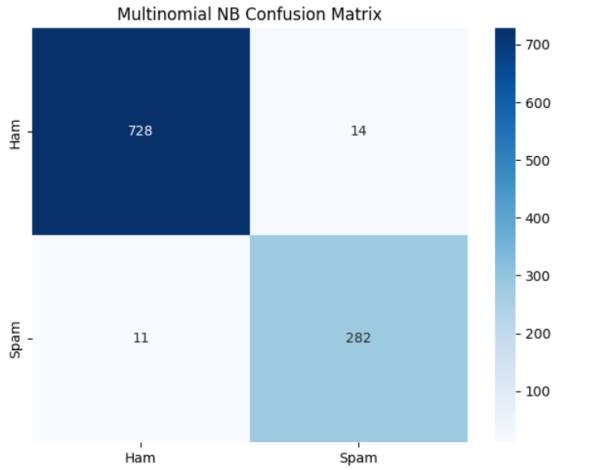
In the following table, we compare the evaluation metrics for the three methods and two models we used for this assignment. As you can see, there is slight improvement for label balanced version and no sensible change for additional features.

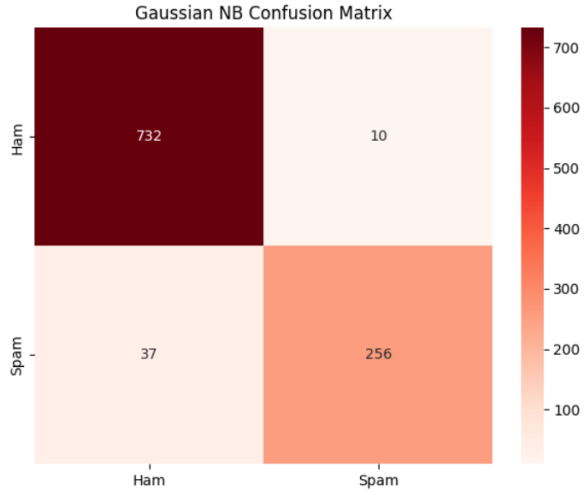
Confusion matrices are also insightful, showing how number of false positives and false negatives, providing further details about model performance. For example, for multinomial NB, there are 11 false positive and 14 false negatives.

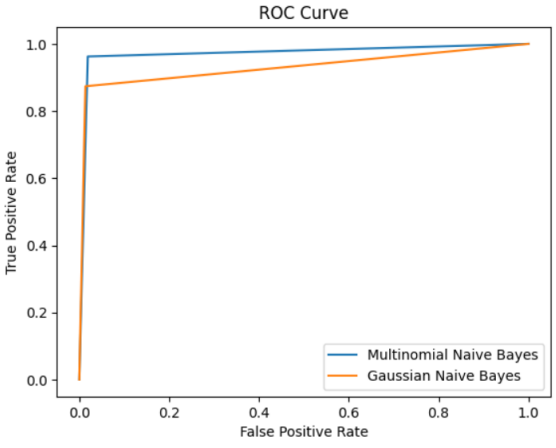
The ROC curve shows how well a classifier performs by plotting the true positive rate (recall) against the false positive rate at different thresholds. This helps visualize how effectively the model distinguishes between positive and negative classes.

TABLE 1. RESULTS COMPARISON BETWEEN METHODS.

Metric	Basic Model	Label Balanced	Additional Features
Accuracy-Multinomial Naive Bayes	97.6%	97.8%	97.4%
Accuracy-Gaussian Naive Bayes	95.5%	96.5%	95.5%
Precision MNB	95.3%	98%	94.6%
Precision GNB	96.2%	98.1%	96.2%
Recall MNB	96.2%	97.3%	96.2%
Recall GNB	87.4%	94.6%	87.4%
F1 Score MNB	95.8%	97.6%	95.4%
F1 Score GNB	92%	96%	92%







CONCLUSION

We created and tested three versions of our spam detection model: a baseline model, a balanced data model, and one with extra contextual features. Balancing the dataset slightly improved performance but adding context-based features did not significantly boost accuracy. Our results confirm that Naïve Bayes classifiers are effective for spam detection and emphasize the importance of data distribution on model performance.

CONTRIBUTIONS

Majid Rostami: Basic Model, Balanced Model, Additional Features, Report Writing
Ayush Jovekar: Basic Model, Report Writing
Adita Karle: Basic Model, Report Writing
Shreyash Waghdhare: Basic Model, Report Writing

Signatures:
Majid Rostami
Ayush Jovekar
Adita Karle
Shreyash Waghdhare