Gaussian Naive Bayes Classification of the Spambase Dataset

CS 445/545: Machine Learning, Spring 2025
Programming Assignment #2

May 18, 2025

1 Introduction

This report presents a Gaussian Naive Bayes approach to classifying spam emails using the UCI Spambase dataset. The dataset contains 57 features, which include word frequencies, character frequencies, and patterns in capitalization. These features aim to distinguish between spam and non-spam (ham) emails. The primary objective was to build a custom Naive Bayes classifier capable of achieving 90%+ accuracy on the test set using carefully engineered preprocessing, robust statistical assumptions, and optional ensemble strategies.

2 Methodology

2.1 Data Preparation

We began by downloading the full Spambase dataset from the UCI Machine Learning Repository. The dataset includes 4,601 total examples, each with 57 numerical features and a binary label indicating spam (1) or not-spam (0). To simulate a realistic test environment, we divided the dataset into a training set and a test set using a stratified 70/30 split, resulting in 3,220 training instances and 1,381 test instances. The class proportions (approximately 39% spam) were preserved in both sets to reflect the original distribution.

2.2 Advanced Preprocessing

Although Naive Bayes does not require standardized features due to its independence assumption, empirical testing revealed that preprocessing significantly improved model performance. We applied the following transformations:

- **Log1p Transformation**: Applied $\log(1 + x)$ to reduce skewness and suppress outlier influence.
- **Robust Scaling**: For each feature, we normalized by subtracting the 25th percentile and dividing by the interquartile range (IQR). This approach maintained robustness to extreme values.

- **Outlier Clipping**: Each normalized feature was capped at ±3 IQRs to reduce the influence of extreme feature values.

These steps transformed the dataset into a more Gaussian-like structure, aligning better with the core assumptions of the Gaussian Naive Bayes model.

2.3 Model Implementation

The Gaussian Naive Bayes classifier was implemented from scratch. The main components of the model are:

1. **Class Priors**: Computed as the proportion of spam vs. not-spam emails in the training set, optionally with Laplace smoothing.
2. **Gaussian Likelihood**: For each feature and class, the mean and standard deviation were computed. Features with zero standard deviation were assigned a small value (e.g., 10^{-6}) to avoid divide-by-zero errors.
3. **Log Probability Inference**: To avoid numerical underflow, we computed the log of the posterior probabilities instead of the product of likelihoods.
4. **Feature Importance and Selection**: The model calculated the discriminative power of each feature using a pooled variance-adjusted signal-to-noise ratio. We then selected the top k features for model refinement.

The final class prediction was made using:

$$\text{class}_{NB}(x) = \arg \max_{c \in \{0,1\}} \left[\log P(c) + \sum_{i=1}^{n} \log P(x_i|c) \right]$$

2.4 Ensemble Model

We implemented a lightweight ensemble approach to evaluate the benefits of model diversity. The ensemble contained 7 Naive Bayes classifiers, each with different combinations of:

- Feature subsets (top 20, 25, 30, etc.).
- Prior smoothing values.
- Minimum standard deviation thresholds.

Each model voted on the class prediction, and the final class was selected via majority voting.

2.5 Hyperparameter Optimization

We performed a grid search across several values of:

- `min_std`: $\{10^{-6}, 10^{-5}, 10^{-4}\}$
- `prior_smoothing`: $\{0.0, 0.01, 0.1, 0.5\}$
- `features`: $\{20, 25, 30, 35, 40\}$

The best configuration was selected based on validation accuracy using a hold-out set from the training data.

3 Results

Best Hyperparameters

- `min_std = 1e-5`
- `prior_smoothing = 0.0`
- `features = 25`

Final Performance Metrics

Single Optimized Model:

- Accuracy: **0.9124**
- Precision: **0.8628**
- Recall: **0.9246**
- F1 Score: **0.8926**

Ensemble Model:

- Accuracy: **0.8733**
- Precision: **0.7741**
- Recall: **0.9577**
- F1 Score: **0.8562**

Confusion Matrix (Single Model)

	Predicted Not-Spam	**Predicted Spam**
Actual Not-Spam	757	80
Actual Spam	41	503

Best Performing Model: Single Optimized Naive Bayes

4 Discussion

4.1 Independence Assumption

Naive Bayes assumes conditional independence between features given the class. In the context of spam emails, this assumption is often violated. For instance:

- The word “free” often co-occurs with “money,” “offer,” or “limited time.”
- Capital letters are frequently used with exclamation marks and urgency phrases like “Act Now!”

Despite these correlations, the Naive Bayes model performs well due to its resilience and the use of log probabilities that reduce overestimation from redundant evidence.

4.2 Precision and Recall Trade-off

A key observation was the contrast between the single and ensemble models:

- The single model achieved balanced performance.
- The ensemble model prioritized recall, detecting almost all spam, but misclassified many legitimate emails.

In real-world systems, this trade-off must be tuned:

- High recall is critical when missing spam is unacceptable.
- High precision is necessary when false positives harm user experience.

4.3 Why Naive Bayes Works Despite Assumptions

1. **Log-space computation** mitigates numeric instability and compensates for redundancy.
2. **Overfitting resistance**: Simpler models generalize better on limited data.
3. **Effective decision boundaries**: Even with incorrect probability estimates, the rankings may still be accurate.
4. **Robust to skewed data**: Preprocessing and minimal variance stabilization help the model generalize better.
5. **Scalable to high dimensions**: Naive Bayes works efficiently with many features and small datasets.

4.4 Areas for Further Improvement

- **Multinomial Naive Bayes**: More suitable for word counts than Gaussian models.
- **Threshold tuning**: Varying the classification threshold could improve the precision-recall balance.
- **Second-stage filtering**: A high-recall Naive Bayes classifier could be followed by a high-precision model (e.g., decision tree).
- **Feature Engineering**: Word embeddings or n-grams could capture more context and correlations.

5 Conclusion

The enhanced Gaussian Naive Bayes classifier achieved over 91% accuracy with balanced precision and recall, outperforming both the basic implementation and ensemble variant in overall performance. While the independence assumption is clearly violated in the Spambase dataset, the classifier's simplicity, efficiency, and robustness allow it to remain a strong baseline for spam detection.

Its effectiveness is further amplified through log-space computation, advanced preprocessing, and careful feature selection. Future work could investigate hybrid models that combine Naive Bayes with other techniques for even greater reliability in spam detection applications.