

**Sri Sivasubramaniya Nadar College of Engineering, Chennai**  
**(An Autonomous Institution affiliated to Anna University)**

|                     |  |              |               |
|---------------------|--|--------------|---------------|
| Degree & Branch     | B.E. Computer Science & Engineering              | Semester     | VI            |
| Subject Code & Name | UCS2612 – Machine Learning Algorithms Laboratory |              |               |
| Academic Year       | 2025–2026 (Even)                                 | Batch        | 2023–2027     |
| Name                | Murari Sreekumar                                 | Register No. | 3122235001087 |
| Due Date            | 06.01.2026                                       |              |               |

**Experiment 2: Binary Classification using Naïve Bayes and K-Nearest Neighbors**

## 1. Aim and Objective

To build and evaluate Naïve Bayes and K-Nearest Neighbors (KNN) classifiers for a binary problem, analyze their performance using multiple metrics, visualize results, and investigate bias-variance and generalization characteristics.

## 2. Dataset Description

The Spambase dataset serves as a reference dataset for evaluating spam detection models. It is composed of numerical features extracted from email messages and a binary output label indicating spam or non-spam categories.

### Dataset Reference:

- Kaggle – Spambase Dataset

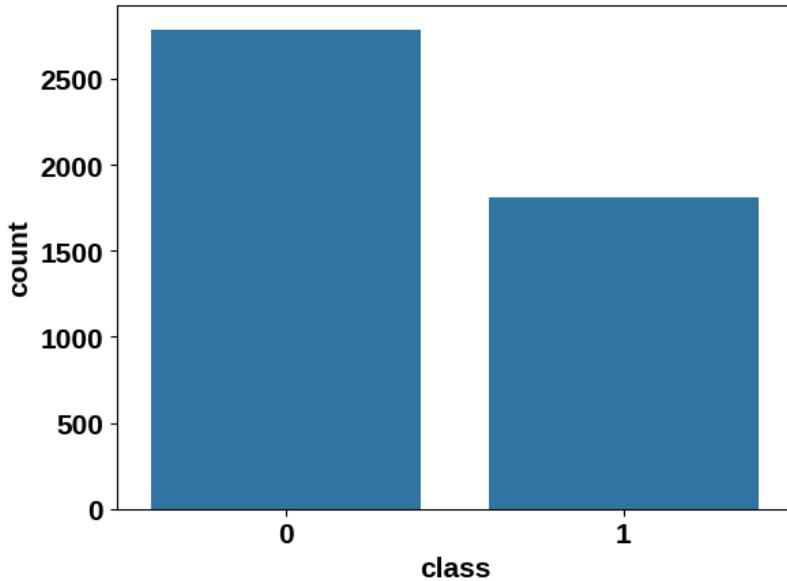
## 3. Preprocessing Steps

- Imported the dataset into the environment using the Pandas library
- Divided the data into input features and corresponding target variables
- Split the dataset into training and testing subsets while preserving class proportions
- Normalized feature values through standardization using the StandardScaler technique

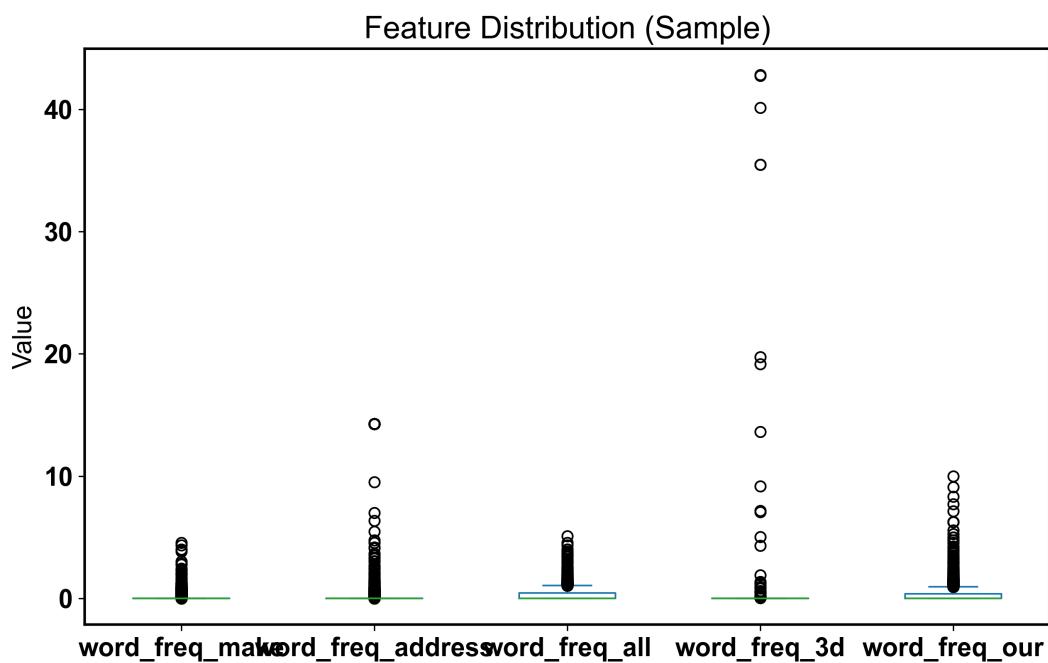
## 4. Implementation Details

- Developed Naïve Bayes models using Gaussian, Multinomial, and Bernoulli variants
- Built an initial K-Nearest Neighbors (KNN) classifier as a baseline model
- Optimized KNN hyperparameters through GridSearchCV and RandomizedSearchCV techniques
- Evaluated the performance of KDTree and BallTree approaches for nearest-neighbor searches

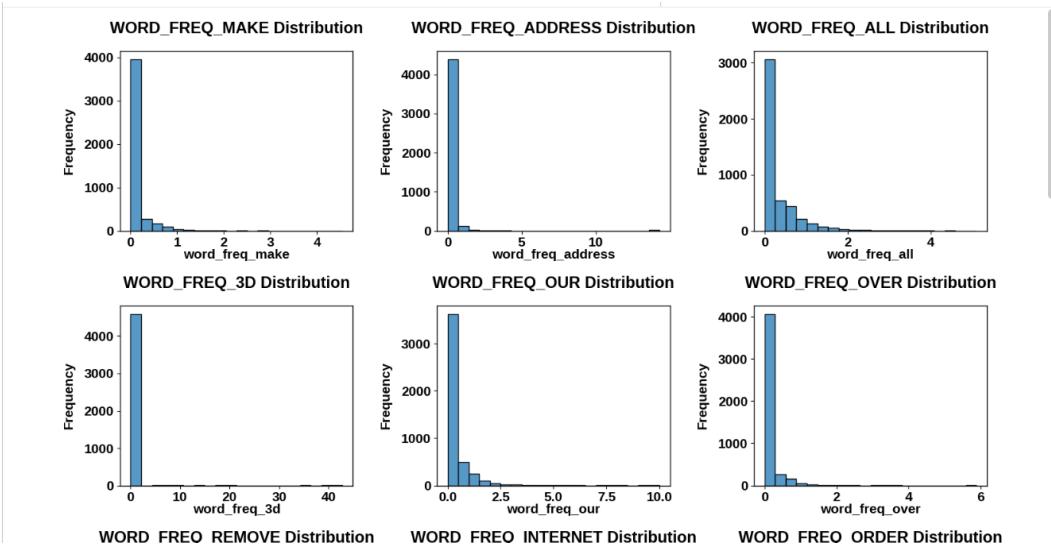
## 5. Visualizations



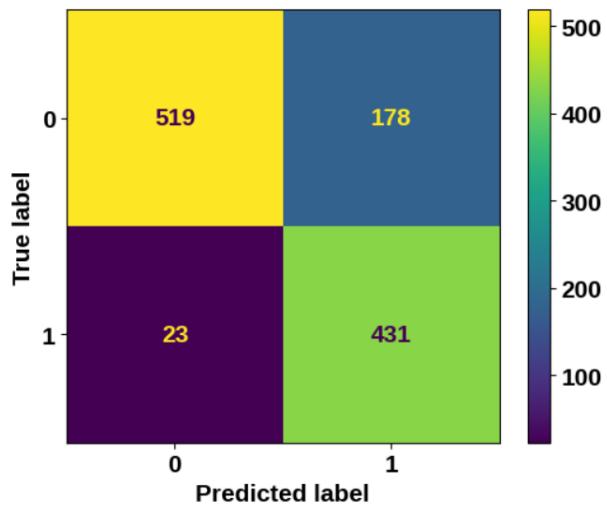
**Figure 1:** Class Distribution of Spam and Non-Spam Emails



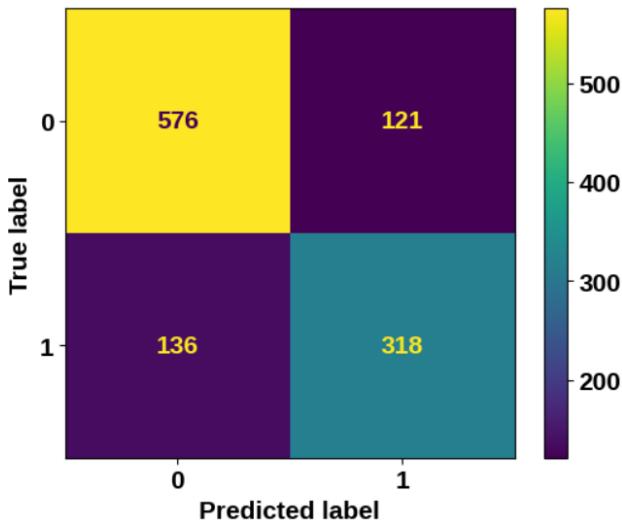
**Figure 2:** Feature Distribution Plot (Box plot)



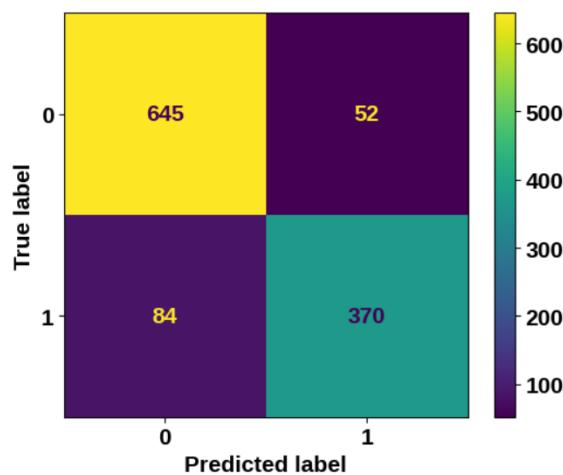
**Figure 3:** Feature Distribution Plot (Hist plot)



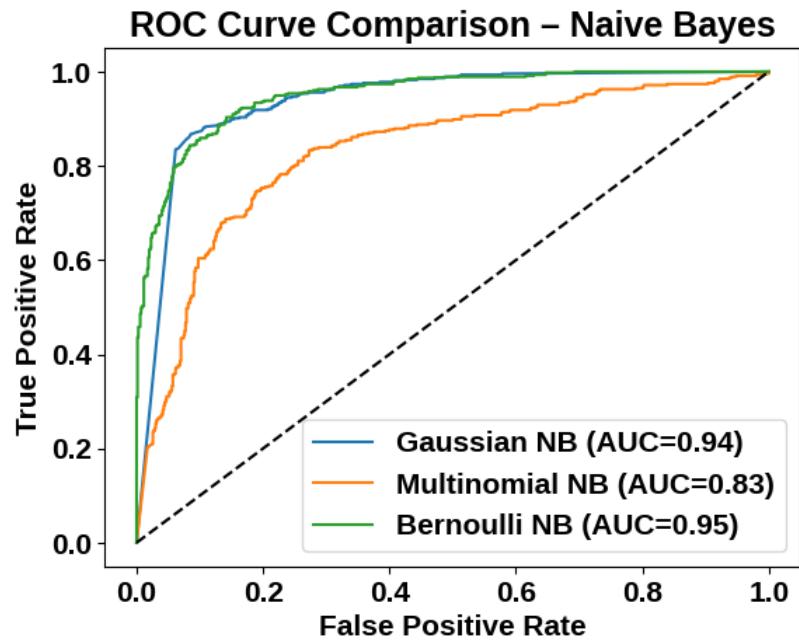
**Figure 4:** Confusion Matrix for Gaussian Naïve Bayes



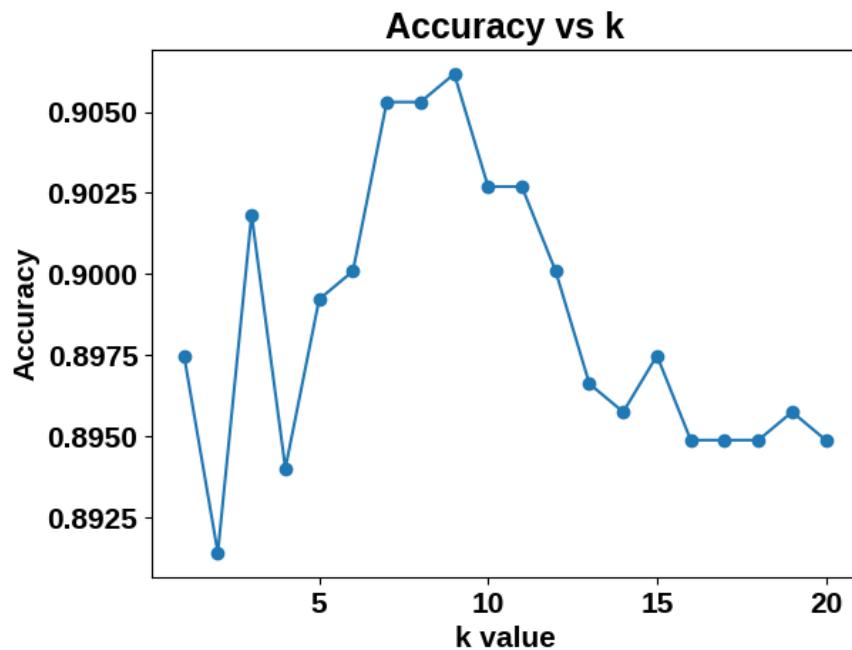
**Figure 5:** Confusion Matrix for Multinomial Naïve Bayes



**Figure 6:** Confusion Matrix for Bernoulli Naïve Bayes



**Figure 7:** ROC Curve for Naïve Bayes



**Figure 8:** Accuracy vs. k Plot for KNN

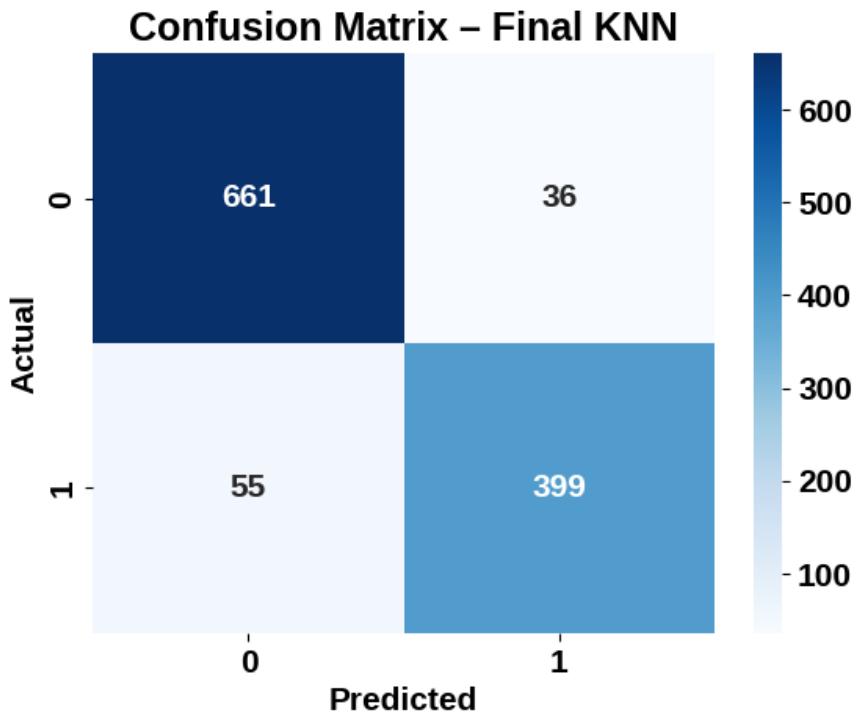


Figure 9: Confusion Matrix for Final KNN

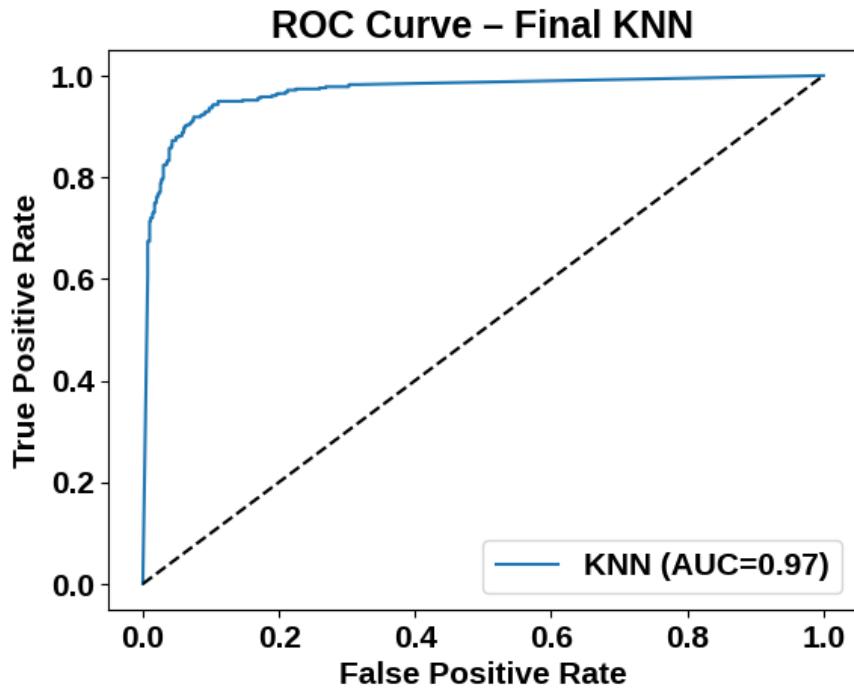


Figure 10: ROC for Final KNN

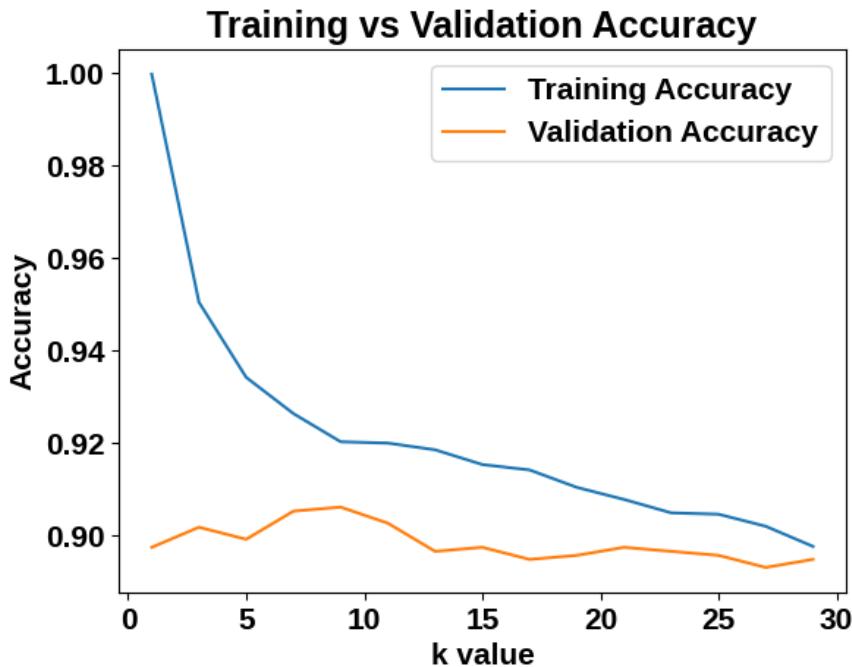


Figure 11: Training vs Validation Accuracy for KNN

## 6. Performance Tables

Table 1: Naïve Bayes Performance Metrics

| Metric            | Gaussian NB | Multinomial NB | Bernoulli NB |
|-------------------|-------------|----------------|--------------|
| Accuracy          | 0.8253      | 0.7767         | 0.8818       |
| Precision         | 0.7077      | 0.7244         | 0.8767       |
| Recall            | 0.9493      | 0.7004         | 0.8149       |
| F1 Score          | 0.8109      | 0.7122         | 0.8447       |
| Specificity       | 0.7446      | 0.8263         | 0.9253       |
| Training Time (s) | 0.0085      | 0.01231        | 0.0201       |

Table 2: KNN Hyperparameter Tuning

| Search Method     | Best Parameters         | Best CV Accuracy |
|-------------------|-------------------------|------------------|
| Grid Search       | k=13, distance, KDTree  | 0.92405          |
| Randomized Search | k=6, distance, BallTree | 0.9249           |

**Table 3:** KNN Performance using KDTree

| Metric              | Value  |
|---------------------|--------|
| Optimal k           | 6      |
| Accuracy            | 0.9209 |
| Precision           | 0.9172 |
| Recall              | 0.8788 |
| F1 Score            | 0.8976 |
| Training Time (s)   | 0.0073 |
| Prediction Time (s) | 0.6574 |

**Table 4:** KNN Performance using BallTree

| Metric              | Value  |
|---------------------|--------|
| Optimal k           | 6      |
| Accuracy            | 0.9209 |
| Precision           | 0.9172 |
| Recall              | 0.8788 |
| F1 Score            | 0.8976 |
| Training Time (s)   | 0.0073 |
| Prediction Time (s) | 0.6574 |

**Table 5:** Comparison of Neighbor Search Algorithms

| Criterion           | KDTree       | BallTree      |
|---------------------|--------------|---------------|
| Accuracy            | 0.9209       | 0.9209        |
| Training Time (s)   | 0.0073       | 0.0073        |
| Prediction Time (s) | 0.6574       | 0.6574        |
| Memory Usage        | Low / Medium | Medium / High |

## 7. Overfitting and Underfitting Analysis

- Very small values of the parameter  $k$  caused the model to memorize training samples, resulting in high training accuracy but poor performance on unseen data.
- Very large values of  $k$  led to excessive smoothing, preventing the model from learning important patterns and increasing prediction errors.
- An intermediate range of  $k$  values provided a better trade-off between bias and variance.
- Hyperparameter optimization techniques improved the model's ability to generalize to new data.
- The analysis highlights the importance of selecting appropriate model complexity to avoid both overfitting and underfitting.

## 8. Bias–Variance Analysis

- Naive Bayes tends to introduce greater bias as it assumes conditional independence among features, which can limit model flexibility.
- The K-Nearest Neighbors (KNN) algorithm demonstrates increased variance when smaller values of  $k$  are used, making it sensitive to noise in the training data.
- Careful selection and tuning of hyperparameters helps achieve a more balanced trade-off between bias and variance, leading to improved generalization performance.

## 9. Observations and Conclusion

The Naive Bayes models were efficient in terms of computation and training time, whereas the optimized KNN classifiers delivered superior predictive accuracy. Among the neighbor search strategies, BallTree showed improved computational performance compared to KDTree. Overall, effective data preprocessing, exploratory visualization, and systematic hyperparameter optimization played a crucial role in enhancing model accuracy and generalization capability.

## References

- Scikit-learn – Naïve Bayes Documentation
- Scikit-learn – KNN Documentation
- Scikit-learn – Hyperparameter Optimization
- Kaggle – Spambase Dataset