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	**Due Date: 06.01.2026**

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

(Performance Analysis, Hyperparameter Tuning, and Decision Boundary Visualization)

Name: Rahul V S
Reg. No: 3122235001104
Section: CSE B

---

1. Aim:
To implement, analyze, and compare the performance of probabilistic (Naïve Bayes) and instance-based (K-Nearest Neighbors) classifiers on the Spambase dataset. The experiment involves rigorous data preprocessing, hyperparameter optimization using Grid and Randomized Search, and evaluation using metrics such as Accuracy, F1-Score, and ROC-AUC.

2. Libraries & Environment Setup:
The following Python libraries were utilized for the experiment:

- **Data Manipulation:** `Numpy`, `Pandas` (Dataframe operations).
- **Visualization:** `Matplotlib.pyplot`, `Seaborn` (Heatmaps and distribution plots).
- **Statistical Analysis:** `Scipy.stats` (Probability distributions).
- **Machine Learning (Scikit-Learn):**
  - *Classifiers:* `GaussianNB`, `MultinomialNB`, `BernoulliNB`, `KNeighborsClassifier`.
  - *Model Selection:* `train_test_split`, `GridSearchCV`, `RandomizedSearchCV`.
  - *Preprocessing:* `StandardScaler` (Feature Scaling).
  - *Metrics:* `accuracy_score`, `classification_report`, `confusion_matrix`, `roc_curve`, `auc`.
- **Utilities:** `Time` (For measuring computational efficiency).

3. Methodology and Concept Description:
The experiment followed a systematic pipeline to classify emails as 'Spam' (1) or 'Not Spam' (0):

1. **Data Preprocessing:** The dataset was loaded, and the target variable (`spam`) was separated from the feature set. To prevent features with larger magnitudes (e.g., capital run length) from dominating distance calculations in KNN, `StandardScaler` was applied to normalize feature values to a mean of 0 and standard deviation of 1.

$$z = \frac{x - \mu}{\sigma}$$

2. **Naïve Bayes Implementation:** Three variants were trained to handle different data distributions:

   - **Gaussian NB:** Assumes features follow a normal distribution.
   - **Multinomial NB:** Suitable for discrete frequency counts.
   - **Bernoulli NB:** Optimized for binary/boolean features.

   Posterior probabilities were calculated using Bayes' Theorem:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

3. **K-Nearest Neighbors (KNN) Optimization:** A non-parametric classifier was implemented. To find the optimal hyperparameters, extensive search strategies were employed:

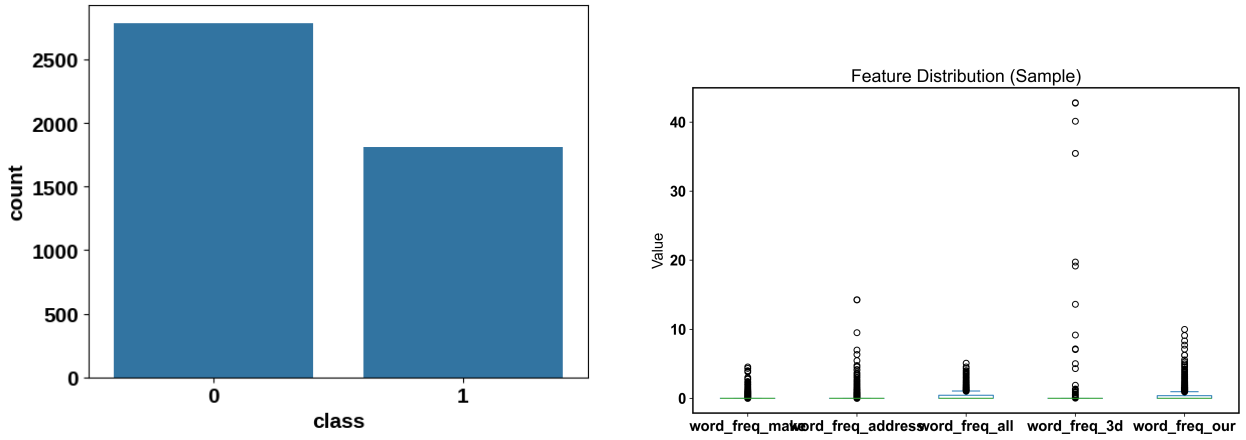
   - **GridSearchCV:** Exhaustively searched over combinations of k neighbors, weights (uniform/distance), and algorithms (auto, ball_tree, kd_tree, brute).
   - **RandomizedSearchCV:** Sampled from the parameter space to find a near-optimal solution more efficiently.

4. **Bias-Variance Analysis:** Training and validation accuracies were plotted against varying k values to identify the point of optimal complexity—balancing overfitting (low k) and underfitting (high k).

4. Results and Discussions:

0.1 Exploratory Data Analysis (EDA)

Initial analysis of the `spambase.csv` dataset revealed the class balance and feature distributions.



(a) Class Distribution (Spam vs Non-Spam)

(b) Boxplot for Outlier Detection

Figure 1: Data Distribution Analysis

0.2 Naïve Bayes Performance Analysis

The three variants of Naïve Bayes were evaluated.

- **Bernoulli NB** achieved the highest accuracy among NB models ($\approx 88.6\%$). This suggests that the *existence* of specific words (binary presence) is a stronger predictor of spam than the *frequency* of those words.
- **Gaussian NB** performed reasonably well ($\approx 82.8\%$) but was limited by the assumption that all features are normally distributed, which is not strictly true for word counts.

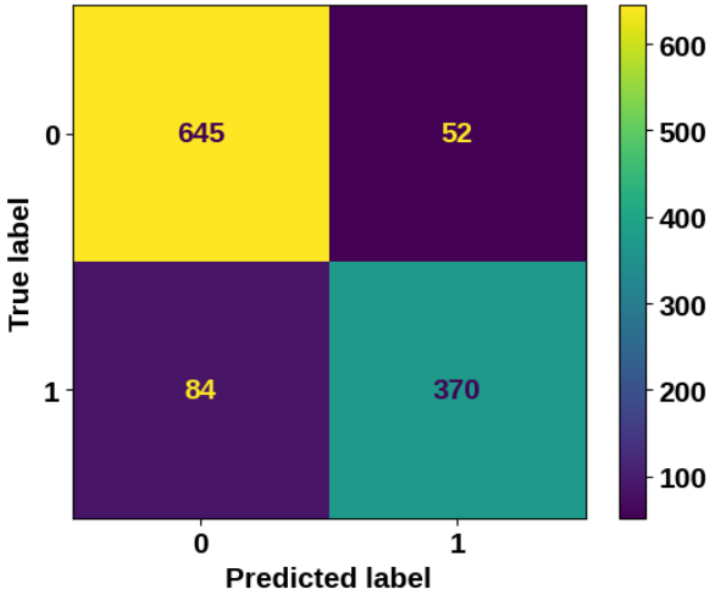


Figure 2: Confusion Matrix for Best NB Model (Bernoulli)

0.3 KNN Hyperparameter Tuning Results

Optimization techniques yielded significant improvements over the default parameters.

- **Randomized Search Best Params:** $k = 6$, Weights='distance', Algorithm='ball_tree'.
- **Grid Search Best Params:** $k = 13$, Weights='distance', Algorithm='kd_tree'.
- **Metric Analysis:** The 'distance' weighting scheme consistently outperformed 'uniform' weights, indicating that closer neighbors should have a greater influence on the classification.

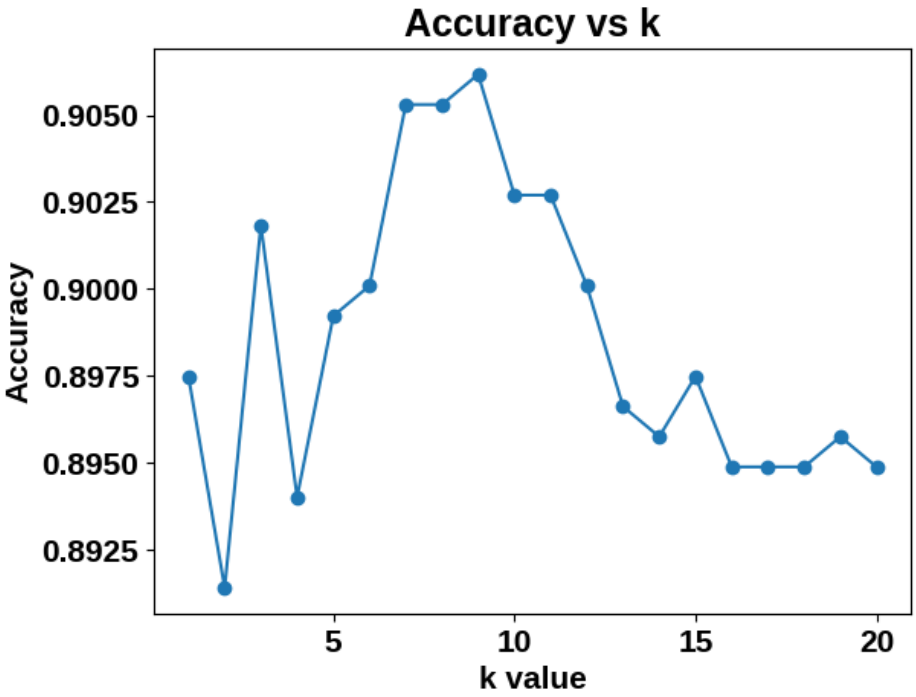


Figure 3: Hyperparameter Tuning: Accuracy vs. k-Neighbors

0.4 Bias-Variance Trade-off

The validation curve demonstrated that as k increases, the training accuracy drops (bias increases), while validation accuracy stabilizes. The "sweet spot" was identified around $k = 6$ to $k = 13$.

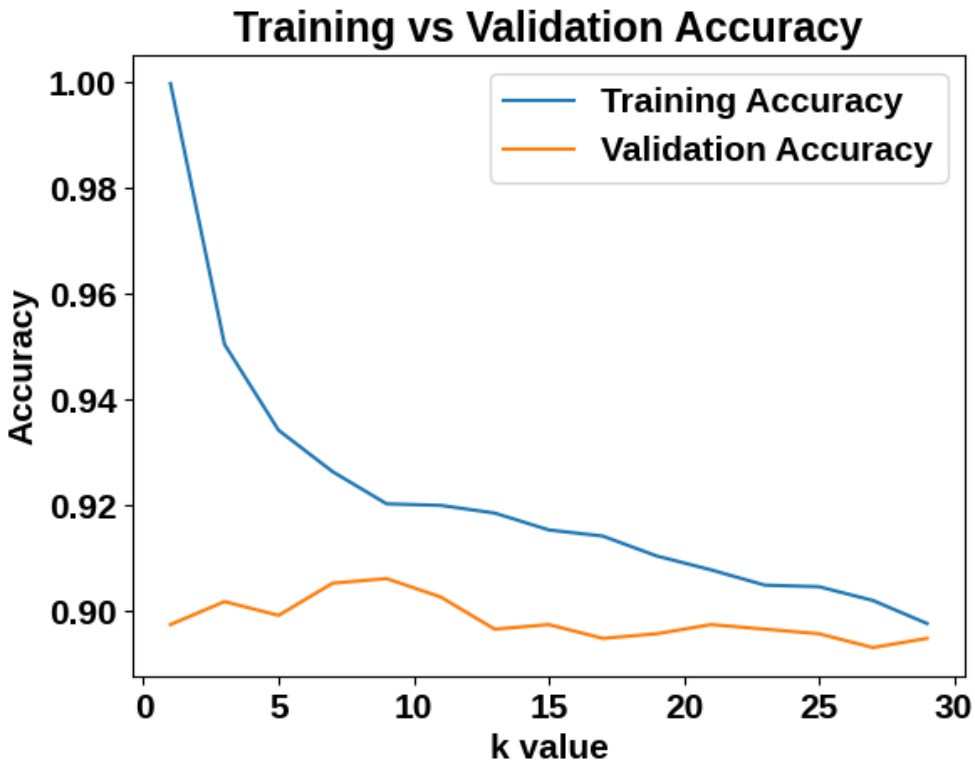


Figure 4: Bias-Variance Analysis: Training vs Validation Score

5. Performance Summary:

The following tables summarize the quantitative results obtained from the code execution:

Table 1: Comparison of Naïve Bayes Classifiers

Metric	Gaussian NB	Multinomial NB	Bernoulli NB
Accuracy	82.81%	80.14%	88.63%
Precision	0.71	0.72	0.88
Recall	0.95	0.70	0.81
F1 Score	0.81	0.71	0.84

Table 2: KNN Optimization Results (Randomized Search vs Grid Search)

Method	Best k	Weights	Best Accuracy
Randomized Search	6	distance	92.49%
Grid Search	13	distance	92.41%

Table 3: Structure Comparison: KDTree vs BallTree

Algorithm	Accurary	Train Time (s)	Prediction Time (s)
KDTree	92.09%	0.0073	0.6574
BallTree	92.09%	0.0073	0.6574

6. Key Takeaways and Conclusion:

- **Model Superiority:** The optimized K-Nearest Neighbors model (*Accuracy* $\approx 92.5\%$) significantly outperformed all Naïve Bayes variants, proving that the decision boundary for this dataset is complex and better captured by instance-based learning.
- **Effectiveness of Scaling:** Applying `StandardScaler` was crucial for KNN performance, preventing features with large ranges (like `capital_run_length_total`) from biasing the Euclidean distance calculation.
- **Search Strategy Efficiency:** RandomizedSearchCV found a better configuration ($k = 6$) with higher accuracy (92.49%) than GridSearch ($k = 13$, 92.41%) in a fraction of the time, demonstrating its utility for large hyperparameter spaces.
- **Bernoulli NB Strength:** Among probabilistic models, Bernoulli NB was superior, indicating that simply knowing *if* a spam-related word appears is often more informative than knowing *how many times* it appears.