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	Mehanth T	Register No.	3122235001080
Due Date	27 January 2026		

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

1. Aim and Objective

To implement Naïve Bayes and K-Nearest Neighbors (KNN) classifiers for a binary classification problem, evaluate them using multiple performance metrics, visualize model behavior, and analyze overfitting, underfitting, and bias–variance characteristics.

2. Dataset Description

A benchmark binary classification dataset containing numerical features and two class labels is used.

Dataset reference:

- Kaggle: Spambase Dataset

The dataset used is the Spambase dataset from the UCI Machine Learning Repository. It consists of 4601 instances, where each instance represents an email. There are 57 continuous real-valued attributes describing the frequency of various words and characters in the email. The target variable is binary: 1 for spam and 0 for non-spam. The dataset is slightly imbalanced, with approximately 39.4% spam and 60.6% non-spam emails.

3. Preprocessing Steps

The preprocessing steps involved:

1. **Missing Value Check**: The dataset was checked for missing values, and none were found.
2. **Data Splitting**: The dataset was split into training (70%) and testing (30%) sets using stratified sampling to maintain class distribution.
3. **Feature Scaling**:
   - **Min-Max Scaling**: Applied for Multinomial Naïve Bayes as it requires non-negative features. Ranges features to [0, 1].
   - **Standard Scaling**: Applied for Gaussian Naïve Bayes and KNN to standardize features (mean=0, variance=1), which is crucial for distance-based algorithms like KNN.

4. Implementation Details

Three variants of Naïve Bayes (Gaussian, Multinomial, Bernoulli) were implemented using Scikit-learn. For K-Nearest Neighbors (KNN), the model was tuned to find the optimal number of neighbors (k). tuning strategies:

- **Grid Search**: Exhaustively searched for the best k in range [1, 30] and evaluated weighting schemes ('uniform' vs 'distance').
- **Randomized Search**: Sampled a subset of the parameter space to find optimal settings efficiently.

Both KDTree and BallTree algorithms were evaluated for neighbor search efficiency.

5. Visualizations

5.1 Exploratory Data Analysis

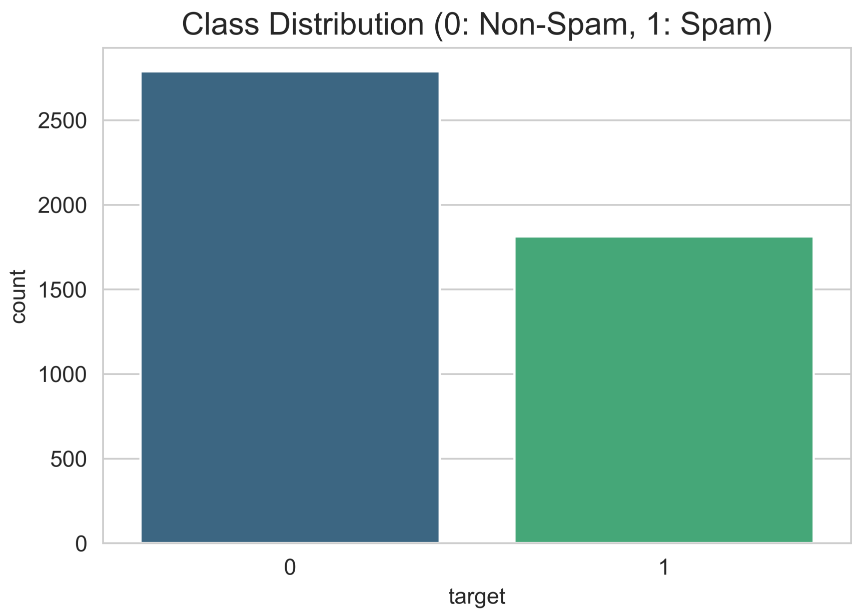

Figure 1: Class Distribution

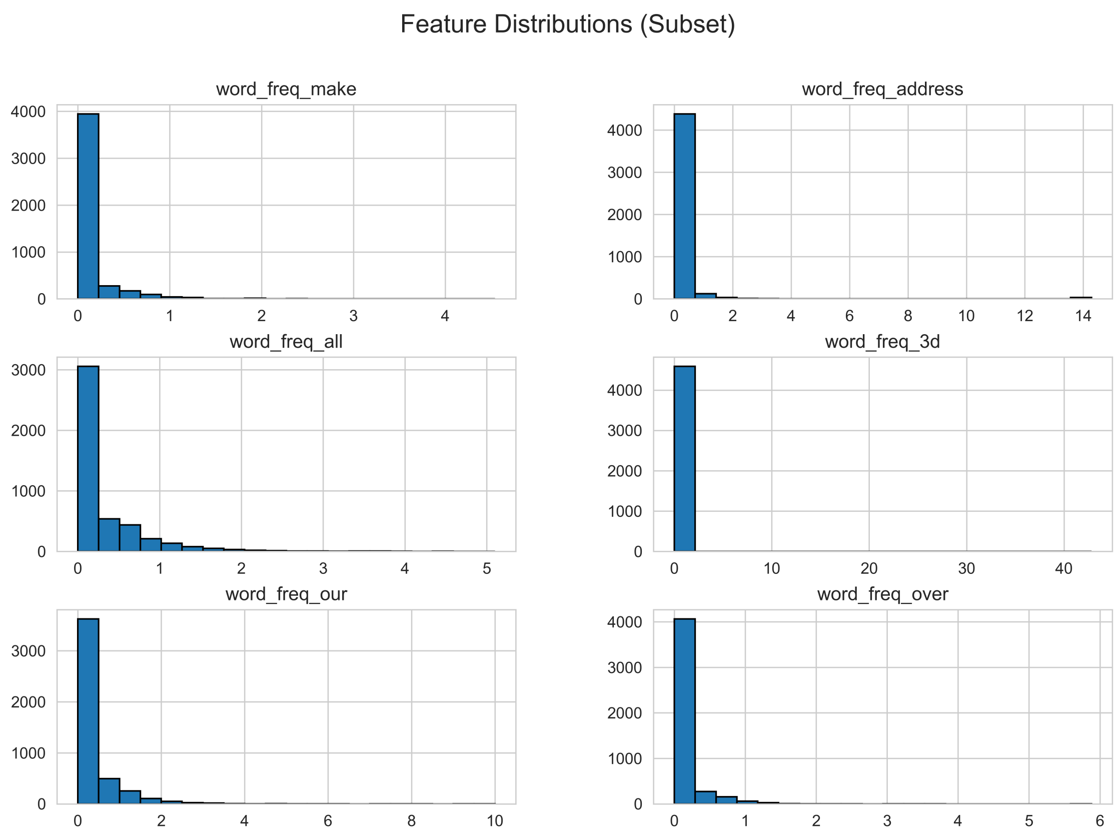

Figure 2: Feature Distributions

5.2 Model Evaluation

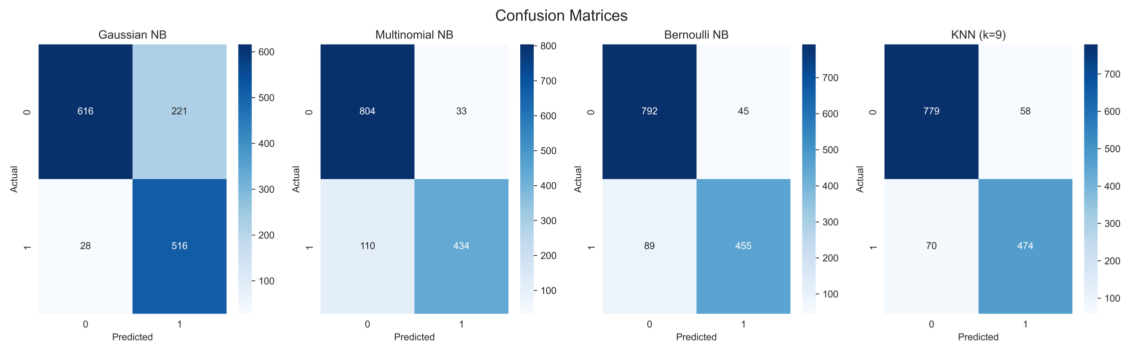

Figure 3: Confusion Matrices for Classifiers

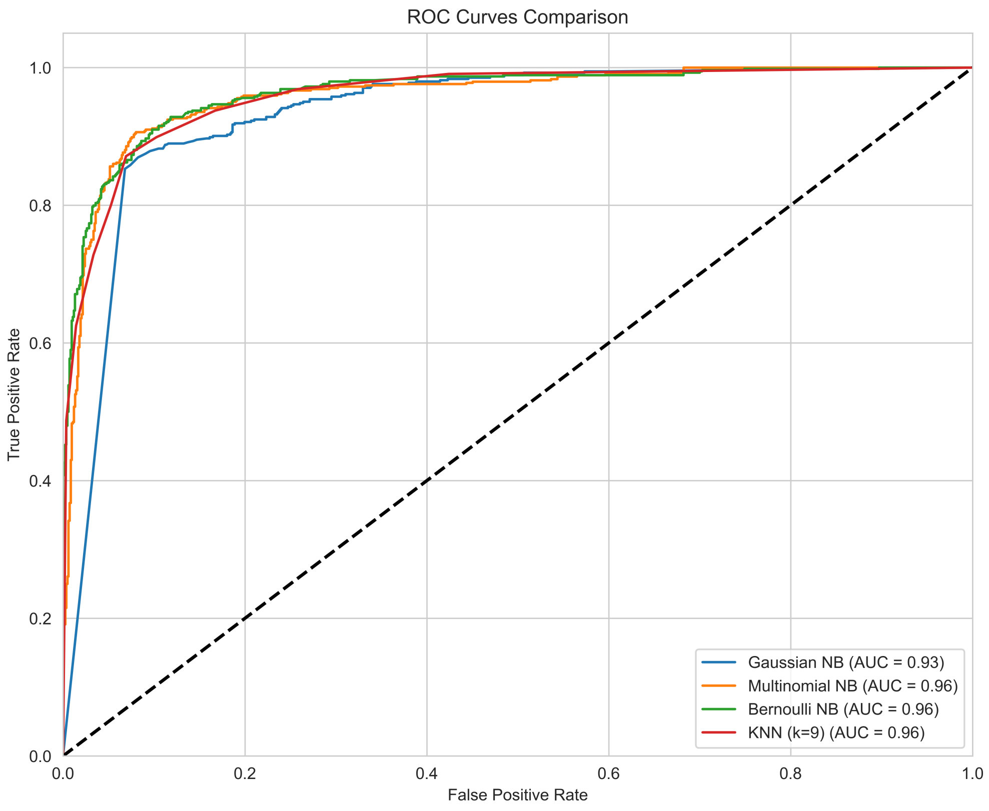

Figure 4: ROC Curves Comparison

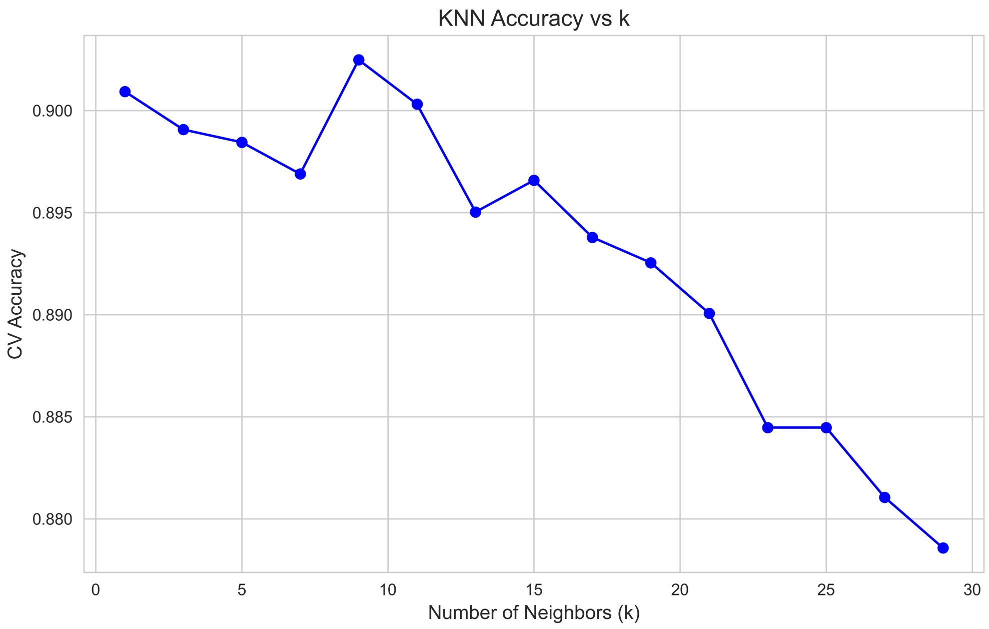

Figure 5: KNN Accuracy vs k

5.3 Bias-Variance Analysis

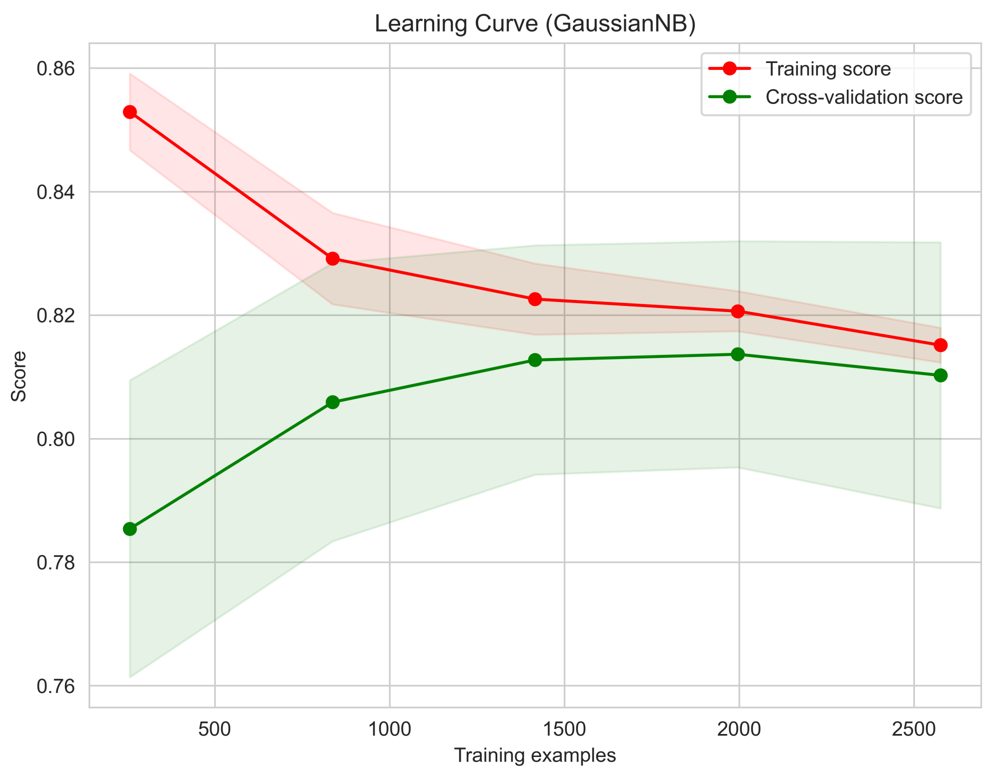

Figure 6: Learning Curve - Naive Bayes

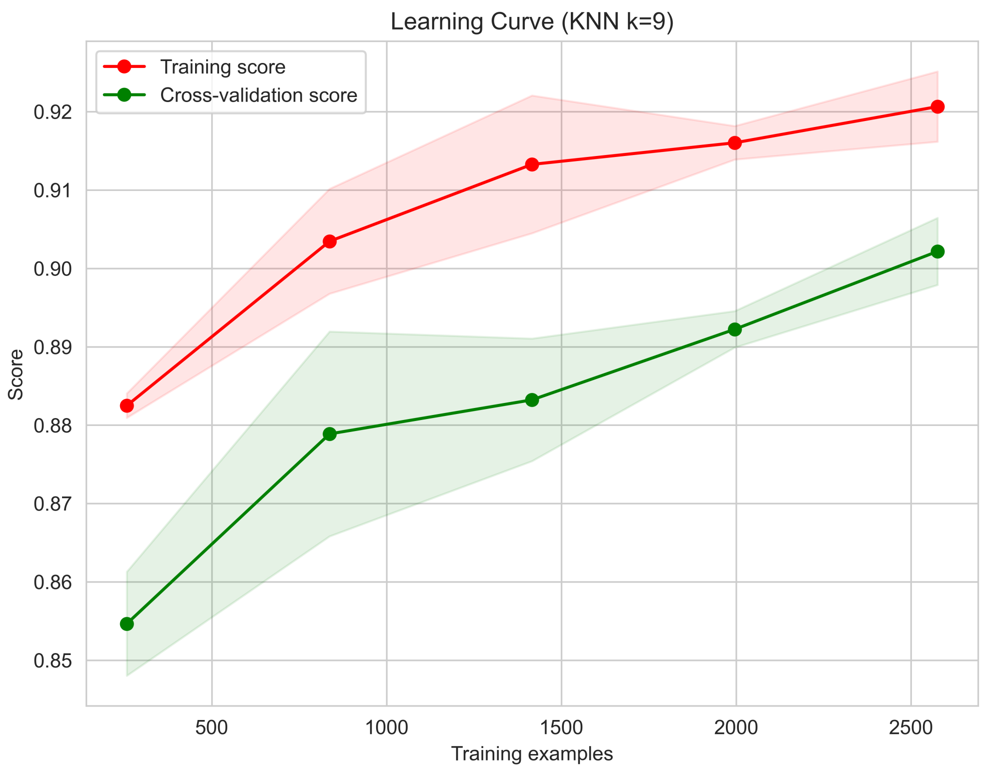

Figure 7: Learning Curve - KNN

6. Performance Tables

Naïve Bayes Performance Comparison

Table 1: Naïve Bayes Performance Metrics

Metric	Gaussian NB	Multinomial NB	Bernoulli NB
Accuracy	0.8197	0.8965	0.9030
Precision	0.7001	0.9293	0.9100
Recall	0.9485	0.7978	0.8364
F1 Score	0.8056	0.8586	0.8716
Specificity	0.7360	0.9606	0.9462
Training Time (s)	0.0033	0.0016	0.0058

KNN Hyperparameter Tuning Results

Table 2: KNN Hyperparameter Tuning

Search Method	Best k	Best CV Accuracy	Best Parameters
Grid Search	9	0.9180	{'n_neighbors': 9, 'weights': 'distance'}
Randomized Search	9	0.9180	{'n_neighbors': 9, 'weights': 'distance'}

KNN Performance using Different Search Methods

Table 3: KNN Performance using KDTree

Metric	Value
Optimal k	9
Accuracy	0.9180
Precision	0.9010
Recall	0.8842
F1 Score	0.8925
Training Time (s)	0.0070
Prediction Time (s)	0.0268

Table 4: KNN Performance using BallTree

Metric	Value
Optimal k	9
Accuracy	0.9180
Precision	0.9010
Recall	0.8842
F1 Score	0.8925
Training Time (s)	0.0054
Prediction Time (s)	0.0388

KDTree vs BallTree Comparison

Table 5: Comparison of Neighbor Search Algorithms

Criterion	KDTree	BallTree
Accuracy	0.9180	0.9180
Training Time (s)	0.0070	0.0054
Prediction Time (s)	0.0268	0.0388
Memory Usage	Low / Medium	Medium / High

7. Overfitting and Underfitting Analysis

The learning curves indicate that KNN with optimal k and distance weighting achieves better generalization than Gaussian Naïve Bayes. As k increases, the model becomes simpler (higher bias), while low k leads to complex boundaries (high variance). The optimal $k = 9$ provides a balance. The small gap between training and validation scores in the learning curve suggests the model is not significantly overfitting.

8. Bias–Variance Analysis

Naïve Bayes (especially Gaussian) typically exhibits higher bias due to its strong independence assumptions, which may not fully hold for word frequencies. However, it requires less training data to converge. KNN is a non-parametric method and typically has lower bias but higher variance (slower convergence of training/validation scores). Hyperparameter tuning helped reduce variance in KNN by selecting an appropriate neighborhood size.

9. Observations and Conclusion

In this experiment, KNN ($k = 9$) outperformed all Naïve Bayes variants in terms of accuracy (91.8%). Among Naïve Bayes models, the Bernoulli and Multinomial variants performed significantly better than Gaussian NB, likely because the word frequency features are better modeled by discrete distributions (after discretization or normalization) than by normal distributions. While KNN offered the best performance, Naïve Bayes was extremely fast in both training and prediction, highlighting the trade-off between accuracy and computational efficiency. KDTree provided faster inference than BallTree for this dataset dimensionality.

References

- Scikit-learn: Naïve Bayes
- Scikit-learn: KNN
- Scikit-learn: Hyperparameter Optimization
- Spambase Dataset