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:	Nanda Kumar B	Roll No:	3122235001701
Due Date	27-01-2026		

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

1. Aim & 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

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

Dataset reference:

- Kaggle: Spambase Dataset

3. Preprocessing Steps

- The dataset (spambase_csv_Kaggle.csv) was loaded into a Pandas DataFrame.
- The dataset was split into the feature matrix (X) and the target vector (y).
- The StandardScaler from the Scikit-Learn library was applied to normalize the feature values.
- The data was divided into training and testing sets for the evaluation of model performance (X_train & X_test & y_train and y_test).

4. Implementation Details

- Implemented Gaussian Naive Bayes, Bernoulli's Naive Bayes and Multinomial Naive Bayes.
- Implemented baseline K-Nearest Neighbors (KNN) Algorithm and with both KD-Tree and Ball-Tree Algorithm.
- Implemented hyperparameter tuning for KNN-Algorithm using the GridSearchCV & RandomizedSearchCV.
- Compared all the algorithm running time and its accuracy score and plotted the various plots to visualize the algorithm implementation.

5. Visualizations

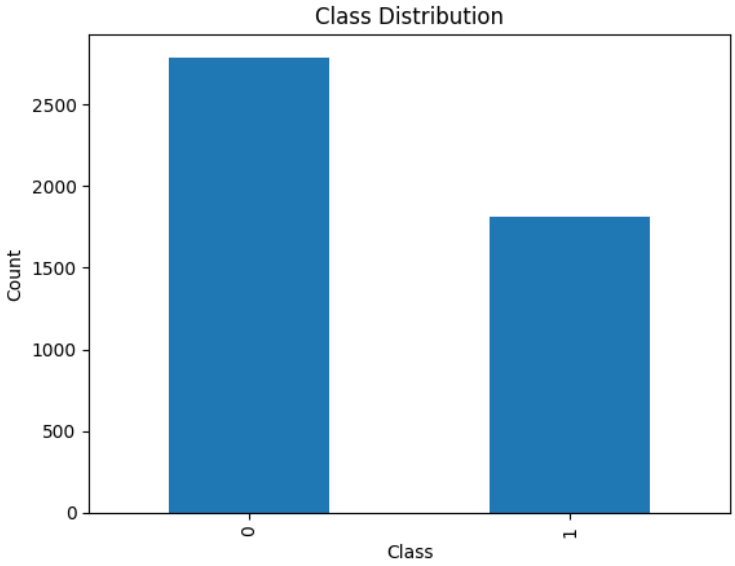

Figure 1: Class Distribution of Spam and Non-Spam Emails

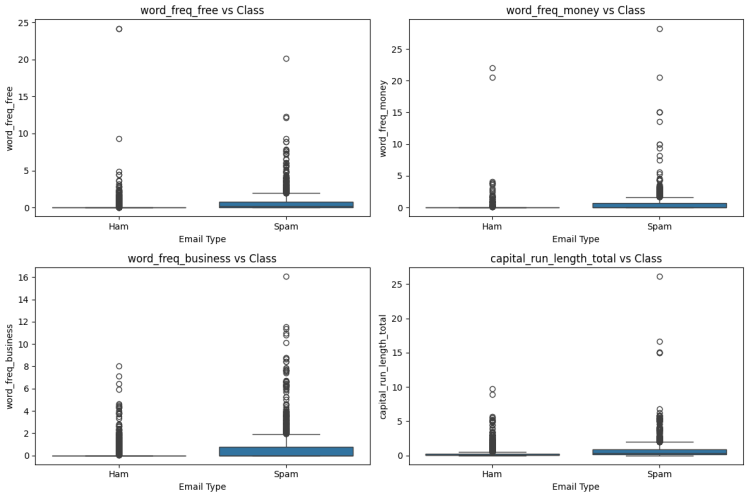

Figure 2: Feature distribution plots

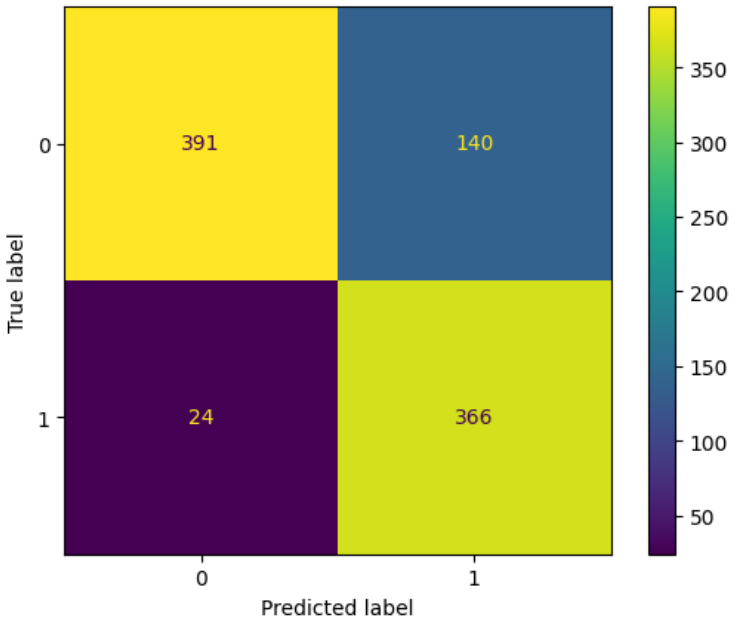

Figure 3: Confusion Matrix for Gaussian Naive Bayes

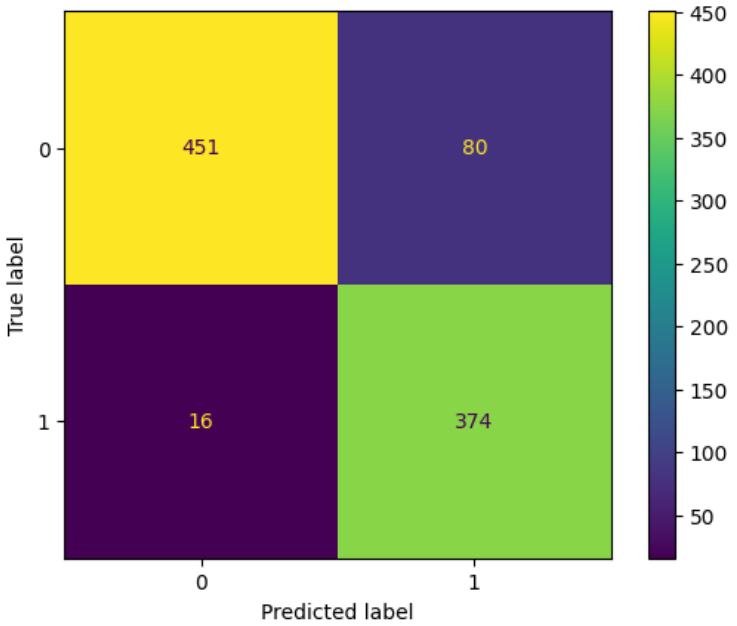

Figure 4: Confusion Matrix for Multinomial Naive Bayes

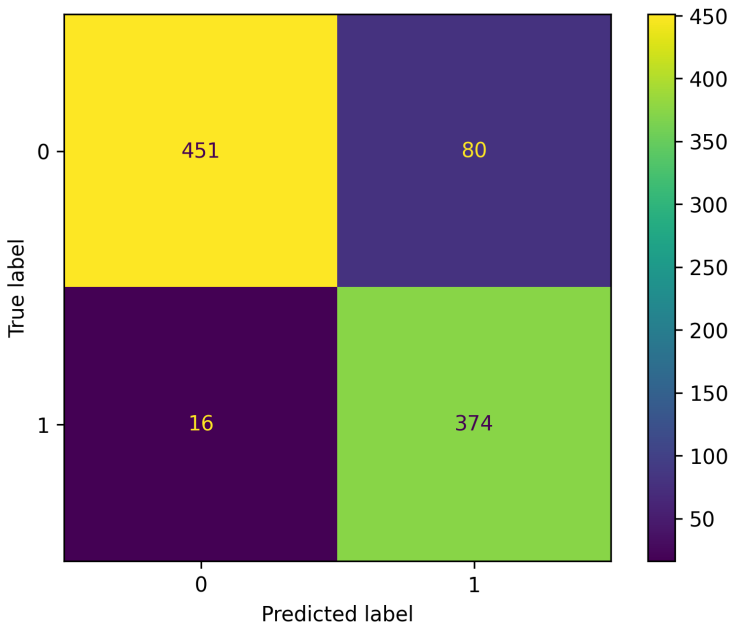

Figure 5: Confusion Matrix for Bernoulli Naive Bayes

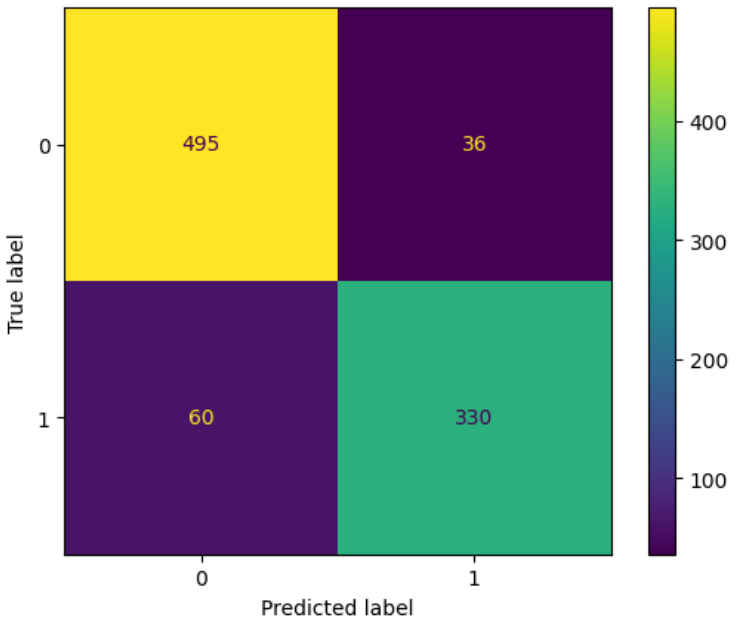

Figure 6: Confusion Matrix for KNN

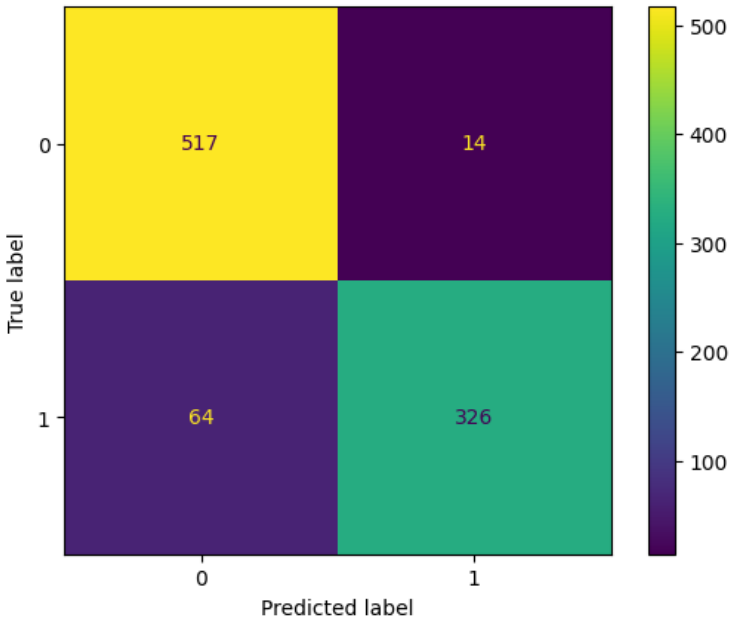

Figure 7: Confusion Matrix for Grid Search KNN

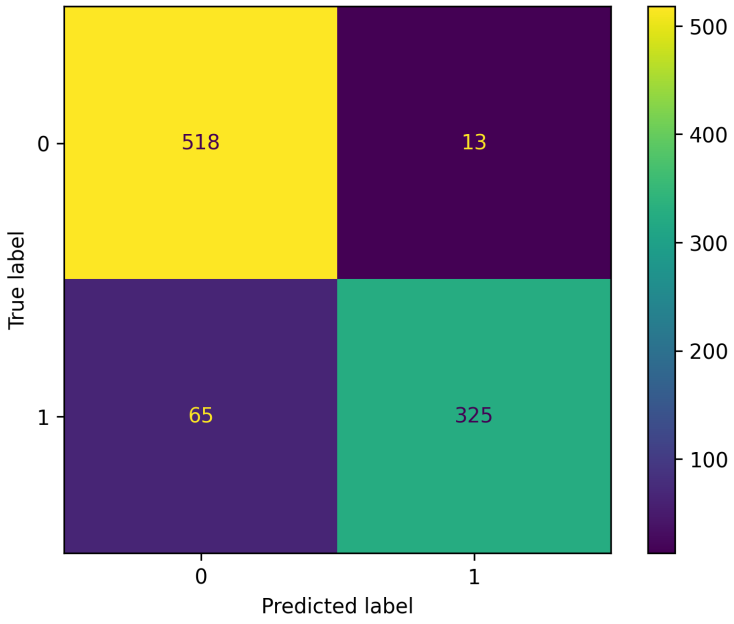

Figure 8: Confusion Matrix for Random Search KNN

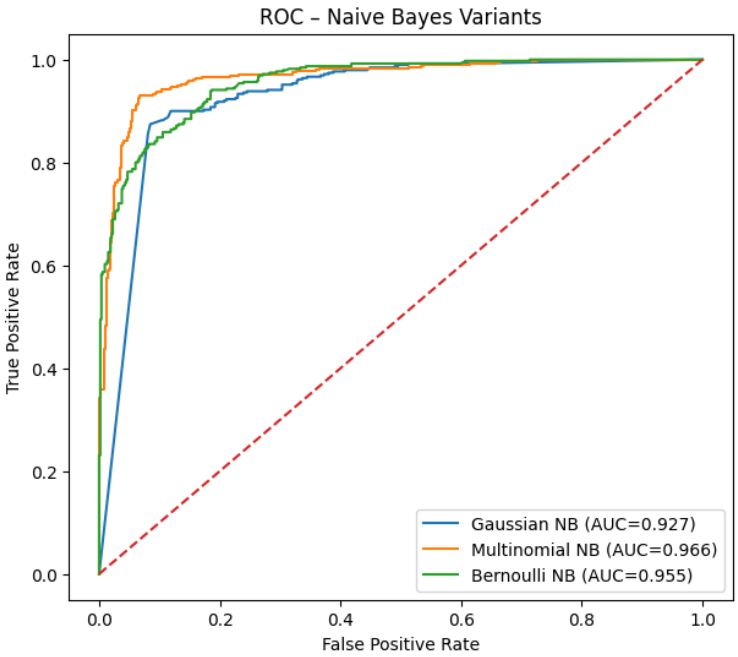

Figure 9: ROC Curve for Naive Bayes Varients

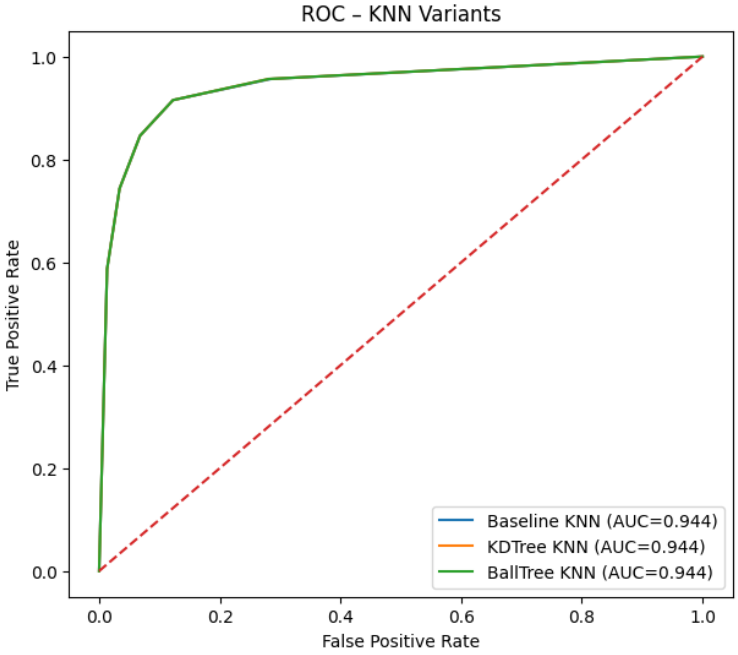

Figure 10: ROC Curve for KNN Varients

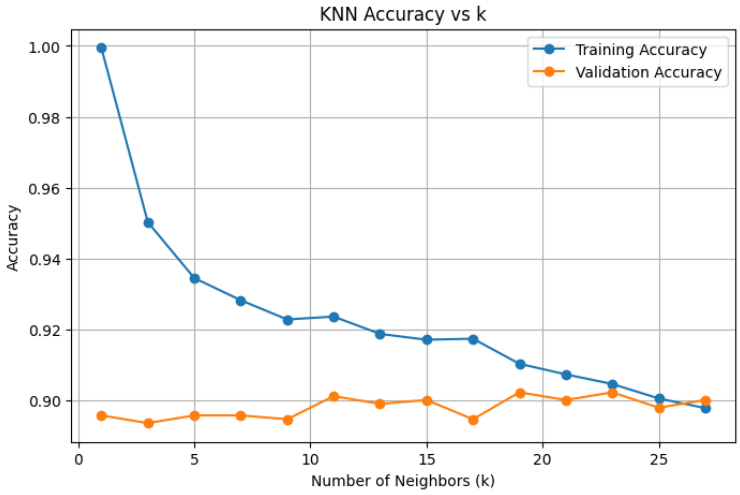

Figure 11: Accuracy vs. k plot for KNN

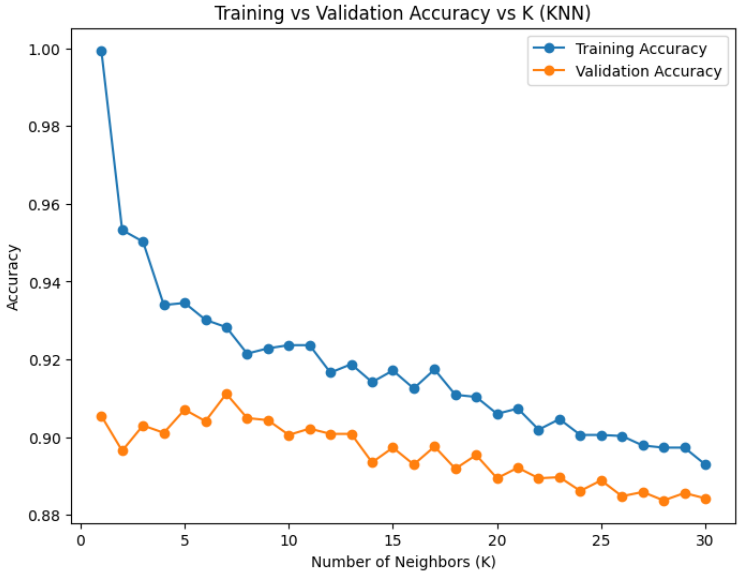

Figure 12: Training vs. validation accuracy plot for KNN

6. Performance Comparison

Table 1: Naïve Bayes Performance Metrics

Metric	Gaussian NB	Multinomial NB	Bernoulli NB
Accuracy	0.82	0.89	0.88
Precision	0.72	0.82	0.90
Recall	0.93	0.95	0.80
F1 Score	0.81	0.88	0.85
Specificity	0.73	0.84	0.93
Training Time (s)	0.00410	0.00109	0.00189

Table 2: KNN Hyperparameter Tuning

Search Method	Best k	Best CV Accuracy	Best Parameters
Grid Search	9	0.92	n_neighbors: 9, p: 1, weights: distance
Randomized Search	11	0.92	n_neighbors: 11, p: 1, weights: distance

Table 3: KNN Performance using BallTree

Metric	Value
Optimal k	5
Accuracy	0.89
Precision	0.90
Recall	0.84
F1 Score	0.87
Training Time (s)	0.015
Prediction Time (s)	0.218

Table 4: KNN Performance using KDTree

Metric	Value
Optimal k	5
Accuracy	0.89
Precision	0.90
Recall	0.84
F1 Score	0.87
Training Time (s)	0.031
Prediction Time (s)	0.291

Table 5: Comparison of Neighbor Search Algorithms

Criterion	KDTree	BallTree
Accuracy	0.89	0.89
Training Time (s)	0.031	0.015
Prediction Time (s)	0.291	0.218
Memory Usage	Low / Medium	Medium / High

7. Overfitting and Underfitting Analysis

- **Difference between training and validation accuracy**
  - For small values of k, the training accuracy is very high, while the validation accuracy is noticeably lower, indicating overfitting. The model fits the training data too closely and does not generalize well.
  - As k increases, the gap between training and validation accuracy reduces, showing improved generalization.
  - For moderate values of k, training and validation accuracies are close and stable, indicating a well-generalized model.
  - For very large values of k, both training and validation accuracies decrease, suggesting underfitting, where the model becomes too simple.
- **Effect of small and large values of k**
  - Small k values (e.g., k = 1, 3):
    - Very high training accuracy
    - Lower validation accuracy
    - High variance and sensitivity to noise
      - Overfitting
  - Medium k values (around the best k found using cross-validation):
    - Balanced training and validation accuracy
    - Best overall performance on unseen data
      - Optimal bias–variance trade-off
  - Large k values:
    - Reduced accuracy on both training and validation sets
    - Decision boundaries become overly smooth
      - Underfitting
- **Role of hyperparameter tuning in generalization**
  - Grid Search and Randomized Search identify an optimal value of k that maximizes cross-validation accuracy.
  - The tuned KNN model shows better validation performance compared to manually chosen k values.
  - This confirms that hyperparameter tuning significantly improves generalization and helps avoid both overfitting and underfitting.
  - Selecting k based only on training accuracy would have resulted in poorer test performance.

8. Bias–Variance Analysis

- **Bias behavior of Naïve Bayes**
  - Naïve Bayes assumes conditional independence among features, which restricts its ability to model complex relationships.
  - Due to this strong assumption, the model shows high bias and low variance.
  - Training and validation accuracies in the notebook are close to each other, indicating stable performance but limited flexibility.
  - This behavior confirms that Naïve Bayes tends to underfit slightly when feature dependencies exist.
- **Variance behavior of KNN**
  - The complexity of the KNN model depends directly on the choice of the neighborhood size k.
  - For small values of k, the model exhibits low bias and high variance, as seen from high training accuracy and fluctuating validation accuracy.
  - As k increases, variance decreases and the model becomes more stable.
  - Moderate k values achieve a balanced bias–variance trade-off, yielding better validation performance.
- **Effect of tuning on bias–variance trade-off**
  - Hyperparameter tuning adjusts the model complexity by selecting an optimal k value.
  - The tuned KNN model reduces excessive variance observed at small k values while avoiding high bias from overly large k values.
  - Proper tuning improves generalization performance and stabilizes validation accuracy.
  - This demonstrates that controlled tuning helps balance bias and variance, whereas aggressive tuning toward very small k can lead to overfitting.

9. Observation & Conclusion

Multiple classification models were evaluated using training and validation metrics to analyze their generalization behavior.Model such as Naïve Bayes exhibited stable performance with minimal variance but were limited by higher bias. In contrast, KNN showed good flexibility, achieving improved accuracy when appropriately tuned, but was prone to overfitting for small neighborhood sizes.

Hyperparameter tuning and cross-validation played a crucial role in identifying optimal configurations and ensuring fair model comparison. The results highlight the importance of balancing bias and variance to achieve robust generalization.

References

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