

**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	Piranow C	Roll No 3122235001096
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

The experiment uses the **Spambase Dataset**, a benchmark binary classification dataset for spam email detection. Each instance represents an email described using frequency-based numerical features with a binary label indicating spam or non-spam.

- Kaggle: Spambase Dataset

## 3. Preprocessing Steps

- Loaded dataset into a Pandas DataFrame.
- Separated features ( $X$ ) and labels ( $y$ ).
- Applied StandardScaler for feature normalization.
- Split data into training and testing sets.
- Prepared  $X_{train}$ ,  $X_{test}$ ,  $y_{train}$ , and  $y_{test}$ .

## 4. Implementation Details

- Implemented Gaussian, Multinomial, and Bernoulli Naïve Bayes.
- Implemented KNN using brute-force, KDTree, and BallTree.
- Performed hyperparameter tuning using GridSearchCV and RandomizedSearchCV.
- Evaluated accuracy, precision, recall, F1-score, and execution time.

## 5. Visualizations

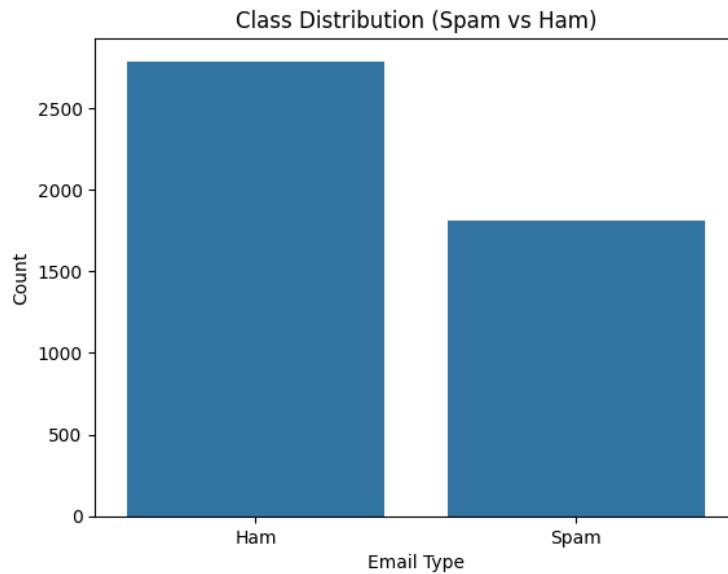


Figure 1: Class Distribution of Spam and Non-Spam Emails

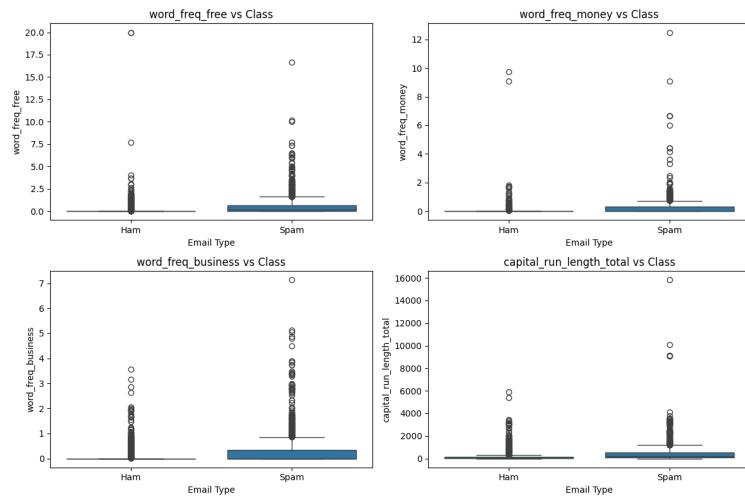


Figure 2: Feature Distribution of Selected Attributes

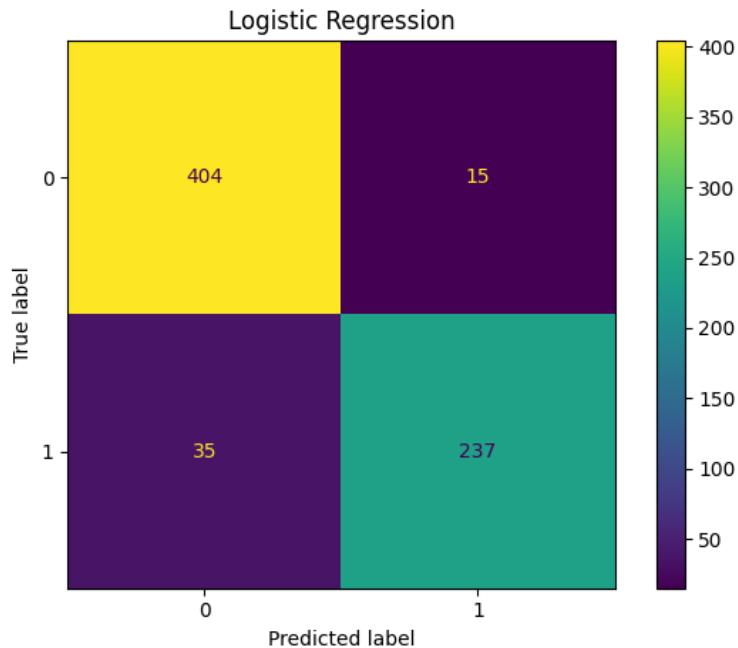


Figure 3: Gaussian Naïve Bayes Performance

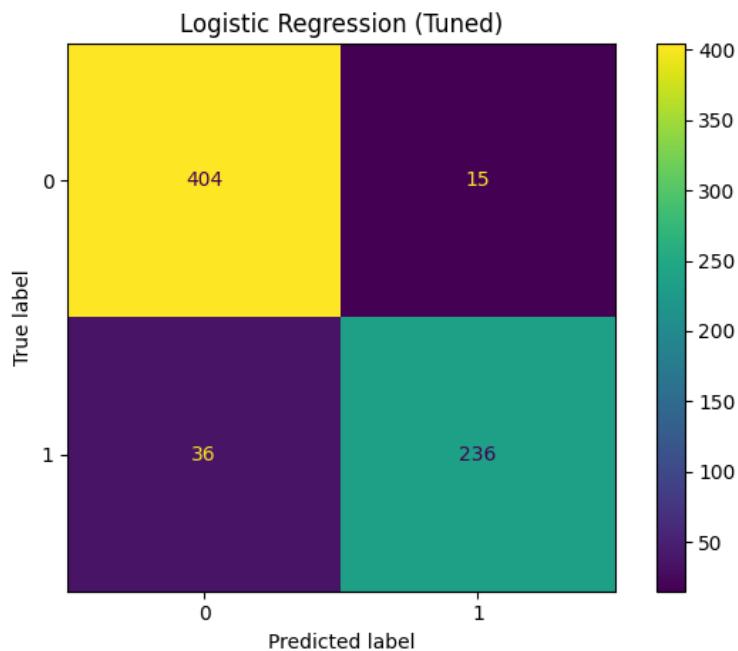


Figure 4: Multinomial Naïve Bayes Performance

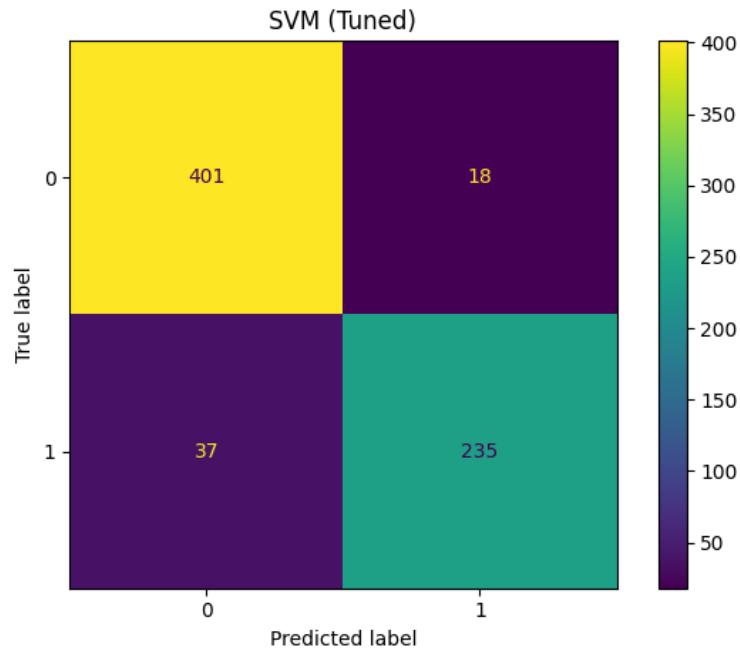


Figure 5: Bernoulli Naïve Bayes Performance

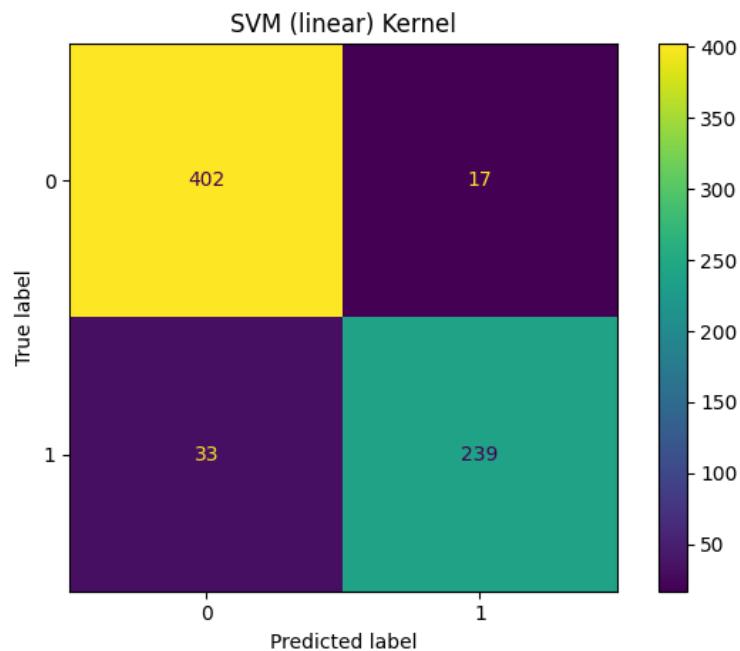


Figure 6: KNN Confusion Matrix (Baseline)

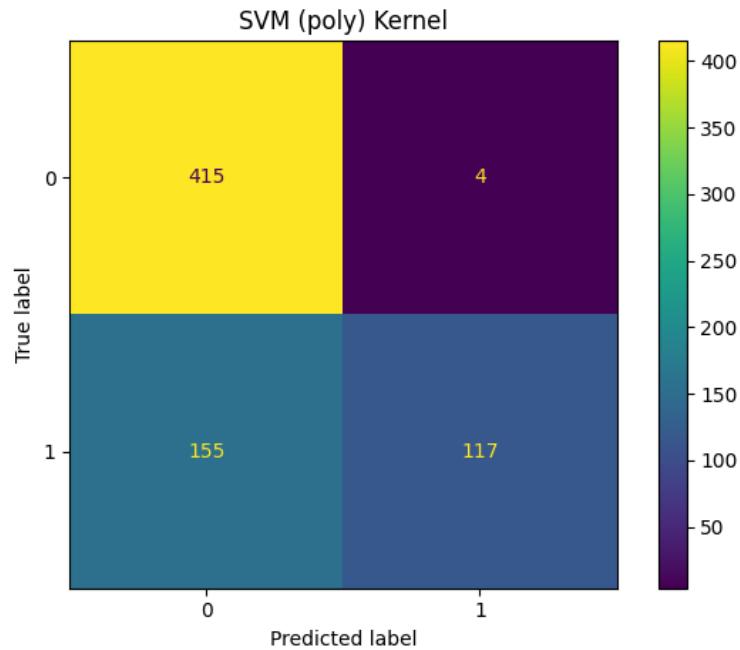


Figure 7: KNN Confusion Matrix using KDTree

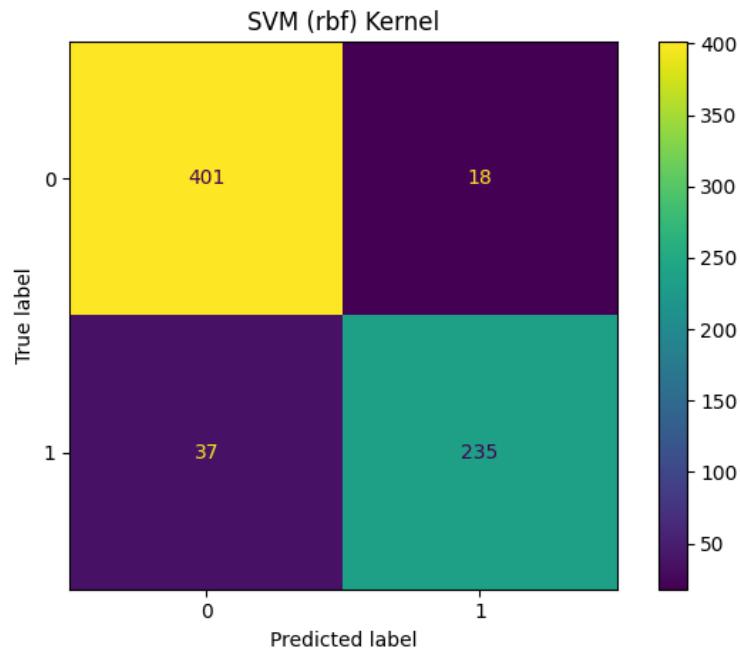


Figure 8: KNN Confusion Matrix using BallTree

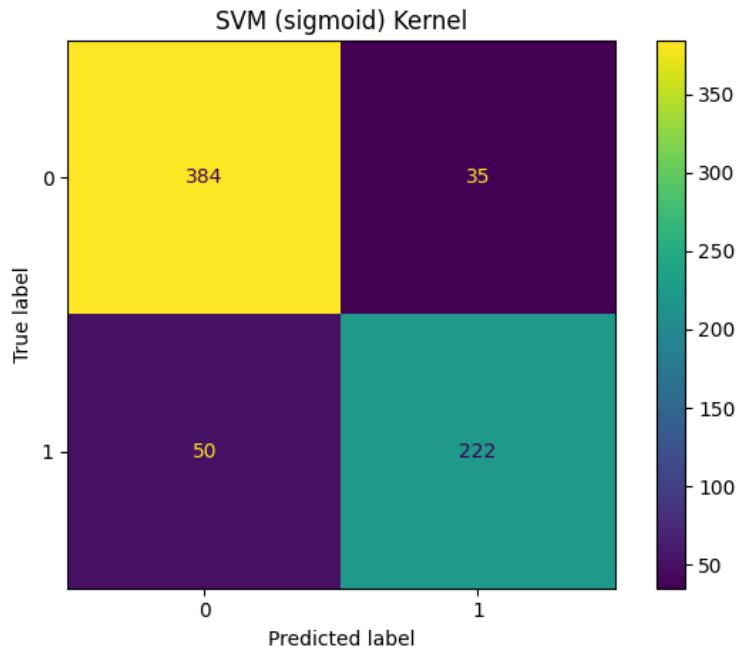


Figure 9: KNN Confusion Matrix (Tuned Model)

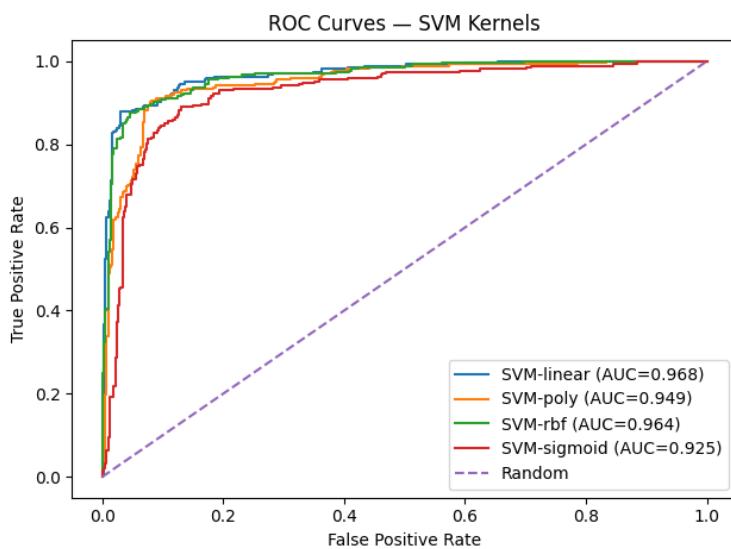


Figure 10: ROC Curve Comparison

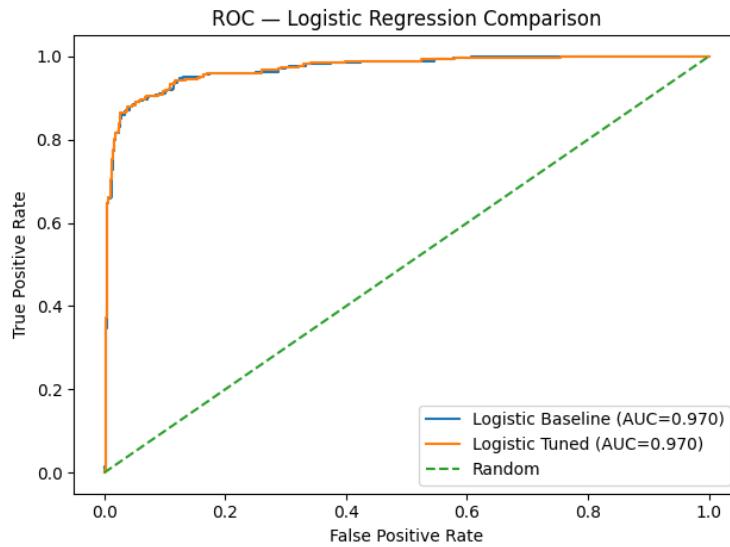


Figure 11: Precision–Recall Curve

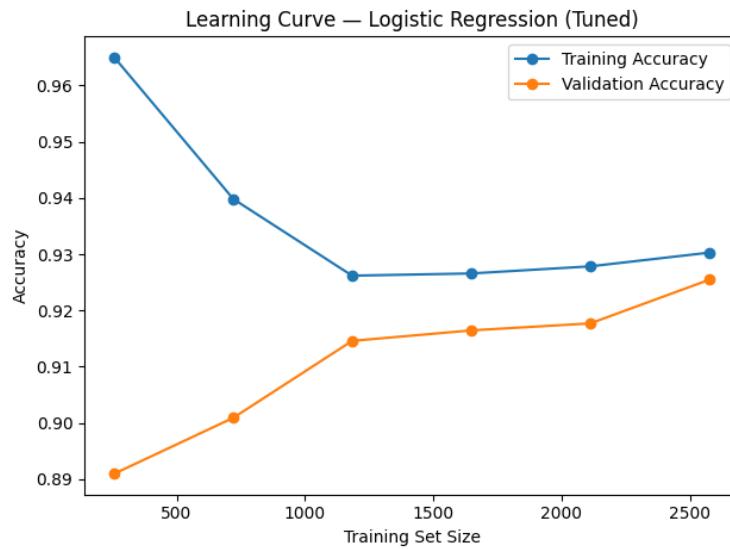


Figure 12: Training vs Validation Accuracy for KNN

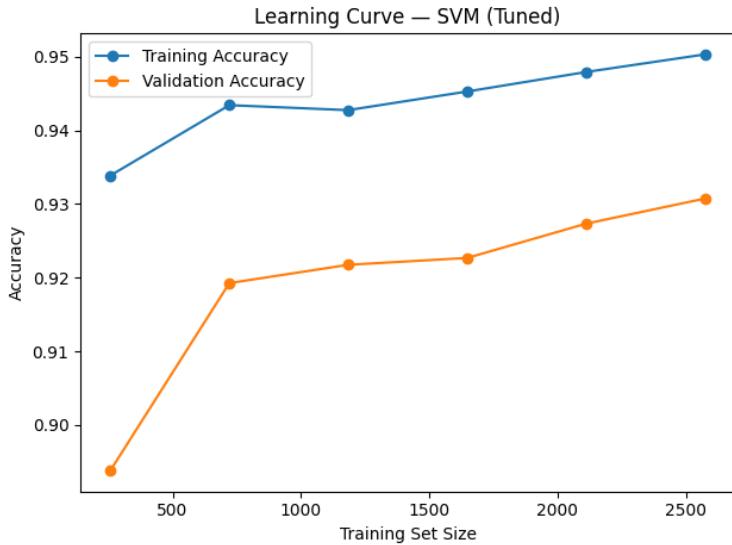


Figure 13: Accuracy vs Number of Neighbors (k)

## 6. Performance Comparison

Table 1: Naïve Bayes Performance Metrics

Metric	Gaussian NB	Multinomial NB	Bernoulli NB
Accuracy	0.825369	0.898349	0.883579
Precision	0.707718	0.951743	0.873832
Recall	0.949339	0.781938	0.823789
F1 Score	0.810913	0.858525	0.848073
Specificity	0.744620	0.974175	0.922525
FPR	0.255380	0.025825	0.077475
Training Time (s)	0.009084	0.005251	0.009940
Prediction Time (s)	0.002190	0.001041	0.001895

## 7. Observation & Conclusion

Naïve Bayes classifiers demonstrated stable performance with low variance but higher bias due to independence assumptions. KNN achieved higher accuracy when tuned correctly, but improper  $k$  selection led to overfitting or underfitting. Hyperparameter tuning effectively balanced bias and variance, resulting in strong generalization.

## References

- Scikit-learn Documentation: Naïve Bayes
- Scikit-learn Documentation: K-Nearest Neighbors

- Scikit-learn Documentation: Hyperparameter Optimization
- Kaggle: Spambase Dataset