Assignment - Classification Algorithm

Problem Statement:

Using patient clinical and laboratory measurements, we want to automatically predict whether a patient has chronic kidney disease (CKD) or not (classification: yes/no).

Problem Identification:

Machine Learning Problem

- Type: Supervised Learning – **Binary Classification**.
- Goal: Build an accurate, reliable classifier that:
    - Correctly identifies CKD patients (high recall for CKD class)
    - Keeps false alarms low (reasonable precision)
    - Has good overall performance (accuracy, F1-score)

Dataset Details:

Total Columns: 25 (Laboratory Measurements)

- Features (24):

  age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hrmo, pcv, wc, rc, htn, dm, cad, appet, pe, ane
- Target (1):

  Classification

Total Rows: 399 (Patient Records)

Feature types:

- **Numeric features:**

    - age: age in years
    - bp: blood pressure
    - al, su: albumin, sugar
    - bgr, bu, sc, sod, pot: various biochemical test results
    - hrmo, pcv, wc, rc: blood count parameters
- **Categorical features:**
    - sg: takes values like a, b, c, d, e
    - rbc, pc: normal, abnormal
    - pcc, ba: present, notpresent
    - htn, dm, cad, pe: yes, no
    - appet, ane: good, poor
- **Target:**
    - classification: yes (CKD) / no (non-CKD)

Data Preprocessing:

Among the input variables, 11 features are **Nominal Data**. To convert this, **One-hot Encoding** technique is used.

- **Binary yes/no columns:**
    - htn, dm, cad, pe, ane, classification: yes → 1, no → 0
- **Binary normal/abnormal:**
    - rbc, pc: abnormal → 1, normal → 0
- **Binary present/notpresent:**
    - pcc, ba: present → 1, notpresent → 0
- **Binary appetite:**
    - appet: poor → 1, good → 0 (if good existed)

One feature has Ordinal Data and **Label Encoding** is used to convert this into numerical values.

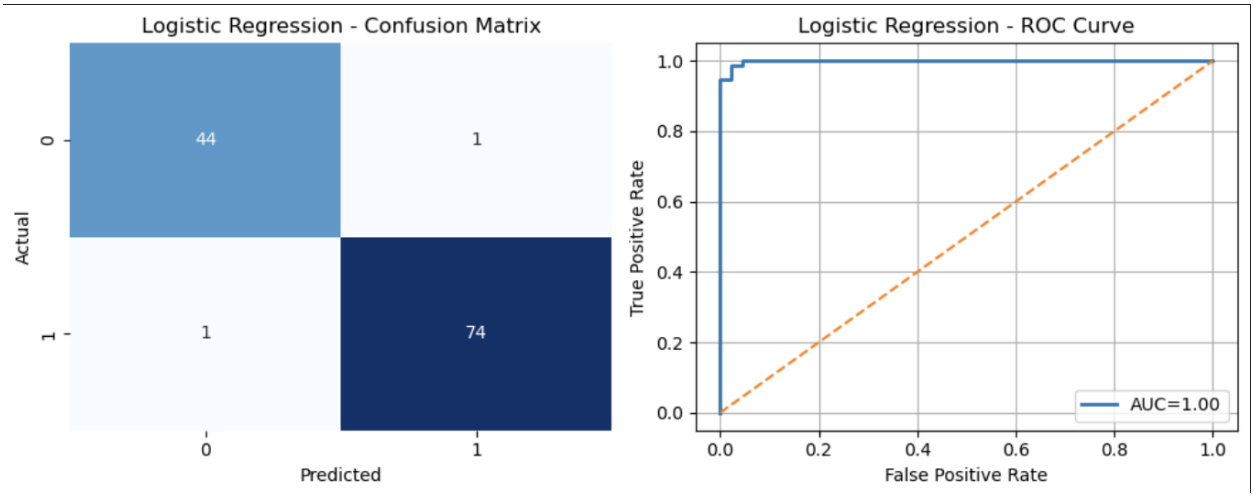
- **ordinal-like feature** sg **(a, b, c, d, e):**
    - Simple **label encoding**: a → 0, b → 1, c → 2, d → 3, e → 4

For **distance-based / linear models** (Logistic Regression, SVM, KNN), StandardScaler is used for **Data Standardization**.

Evaluation metrics across various algorithms for the dataset:

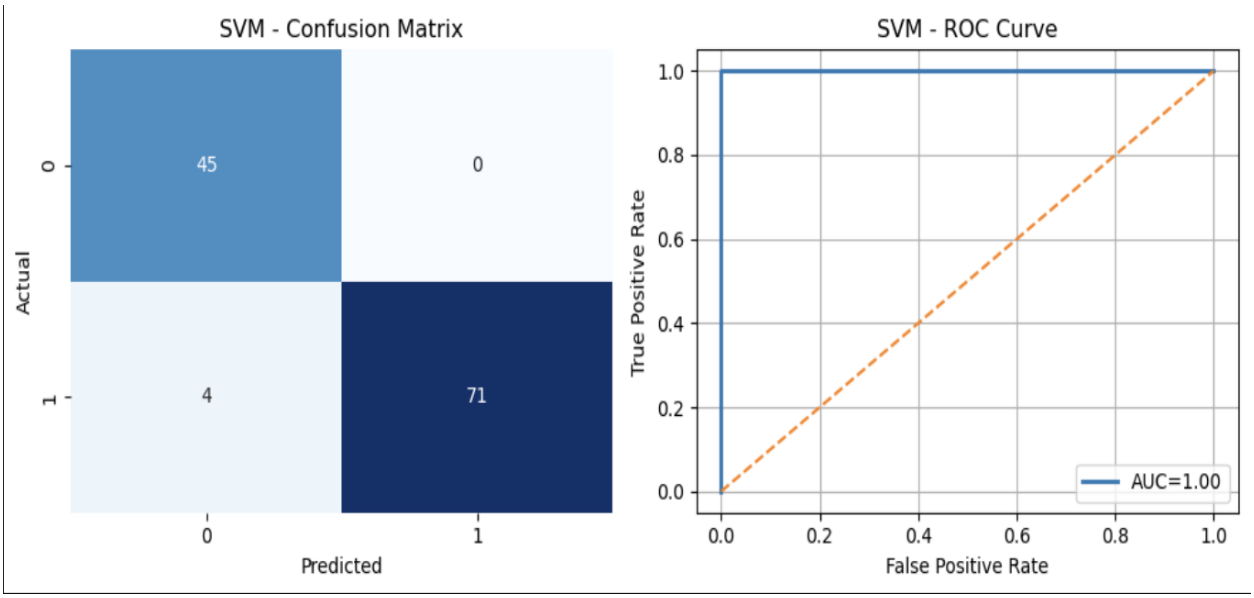
1. Logistic Regression

```
Best Logistic Regression Params: {'penalty': 'l2', 'solver': 'newton-cg'}

=================== Logistic Regression ====================
Accuracy: 0.98
Confusion Matrix:
 [[44  1]
 [ 1 74]]
Classification report:
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        45
           1       0.99      0.99      0.99        75

    accuracy                           0.98       120
   macro avg       0.98      0.98      0.98       120
weighted avg       0.98      0.98      0.98       120

AUC Score: 1.00
```



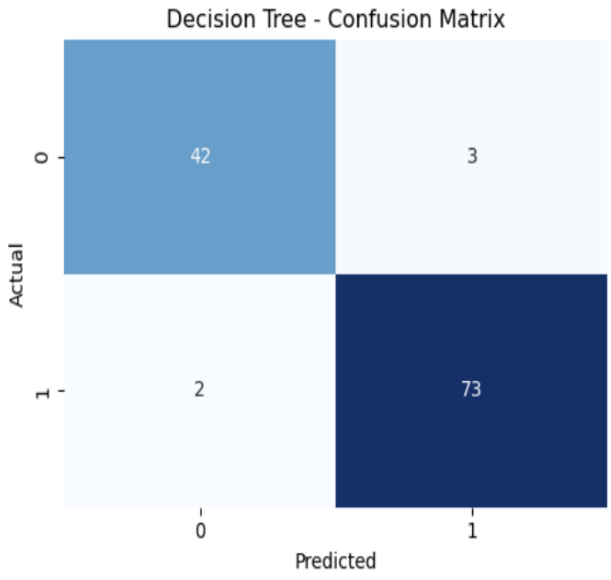
2. Support Vector Machine:

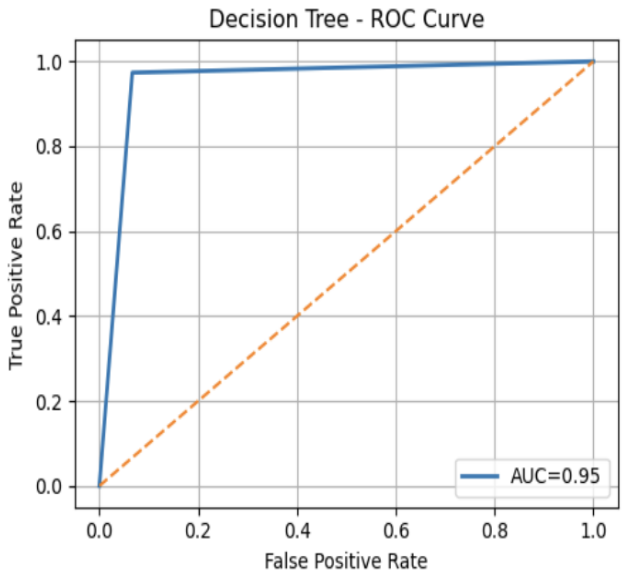
```
Best SVM Params: {'C': 0.1, 'gamma': 'scale', 'kernel': 'rbf'}

==================== SVM ====================
Accuracy: 0.97
Confusion Matrix:
 [[45  0]
 [ 4 71]]
Classification report:
              precision    recall  f1-score   support

           0       0.92      1.00      0.96        45
           1       1.00      0.95      0.97        75

    accuracy                           0.97       120
   macro avg       0.96      0.97      0.97       120
weighted avg       0.97      0.97      0.97       120

AUC Score: 1.00
```



3. Decision Tree:

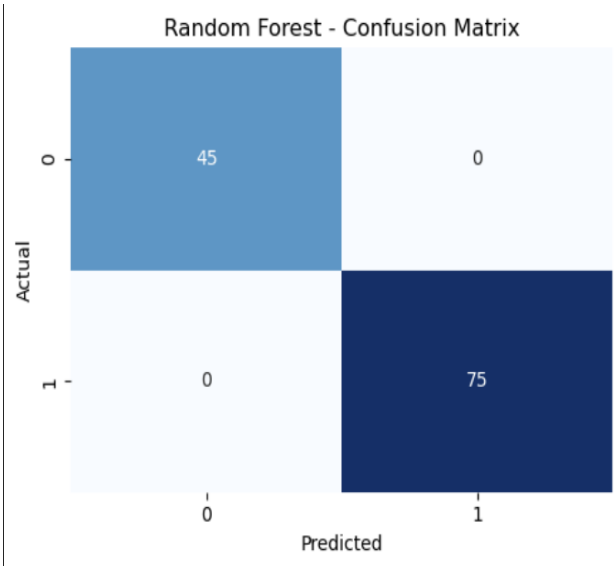
```
Best Decision Tree Params: {'criterion': 'entropy', 'max_features': 'sqrt', 'splitter': 'random'}

==================== Decision Tree ====================
Accuracy: 0.96
Confusion Matrix:
 [[42  3]
 [ 2 73]]
Classification report:
              precision    recall  f1-score   support

           0       0.95      0.93      0.94        45
           1       0.96      0.97      0.97        75

    accuracy                           0.96       120
   macro avg       0.96      0.95      0.96       120
weighted avg       0.96      0.96      0.96       120

AUC Score: 0.95
```

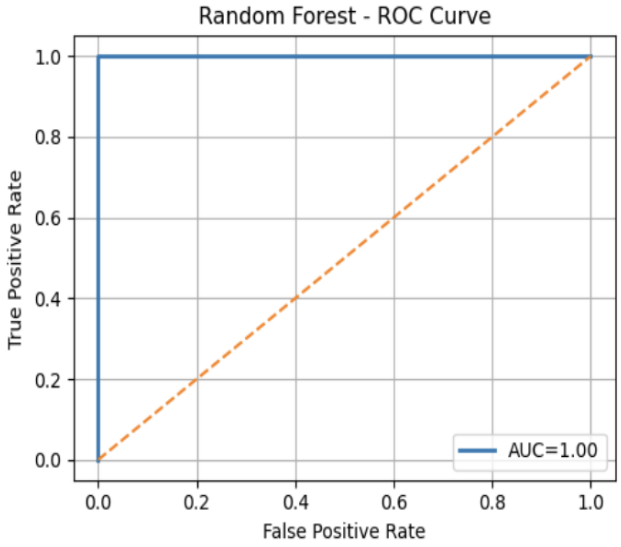




4. Random Forest:

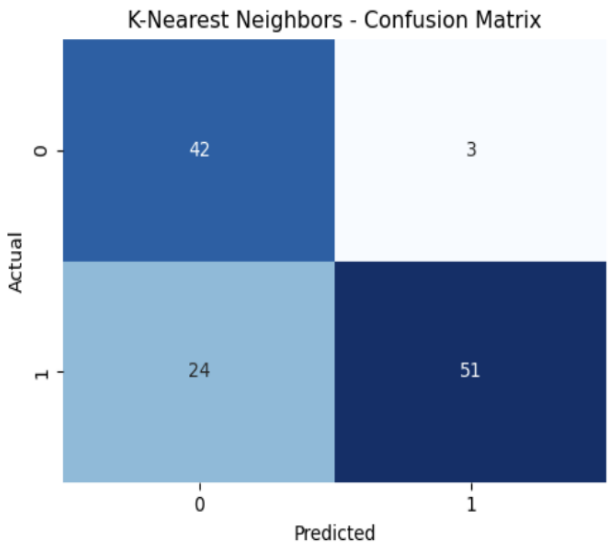
```
Best Random Forest Params: {'criterion': 'gini', 'max_features': 'sqrt', 'n_estimators': 100}

==================== Random Forest ====================
Accuracy: 1.00
Confusion Matrix:
 [[45  0]
 [ 0 75]]
Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        45
           1       1.00      1.00      1.00        75

    accuracy                           1.00       120
   macro avg       1.00      1.00      1.00       120
weighted avg       1.00      1.00      1.00       120

AUC Score: 1.00
```

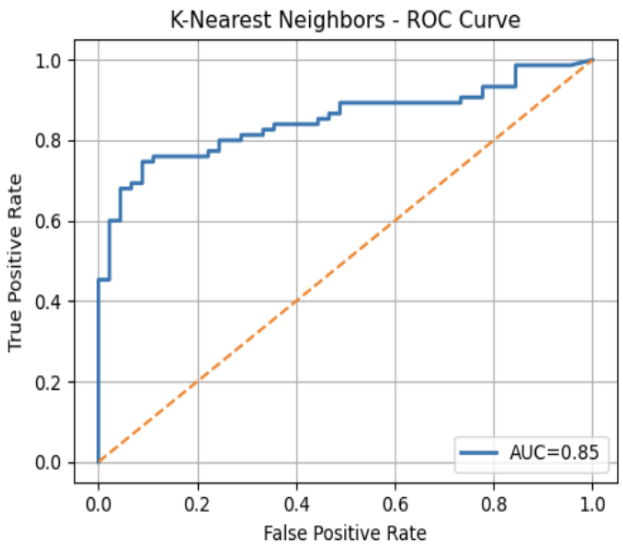




5. K – Nearest Neighbors:

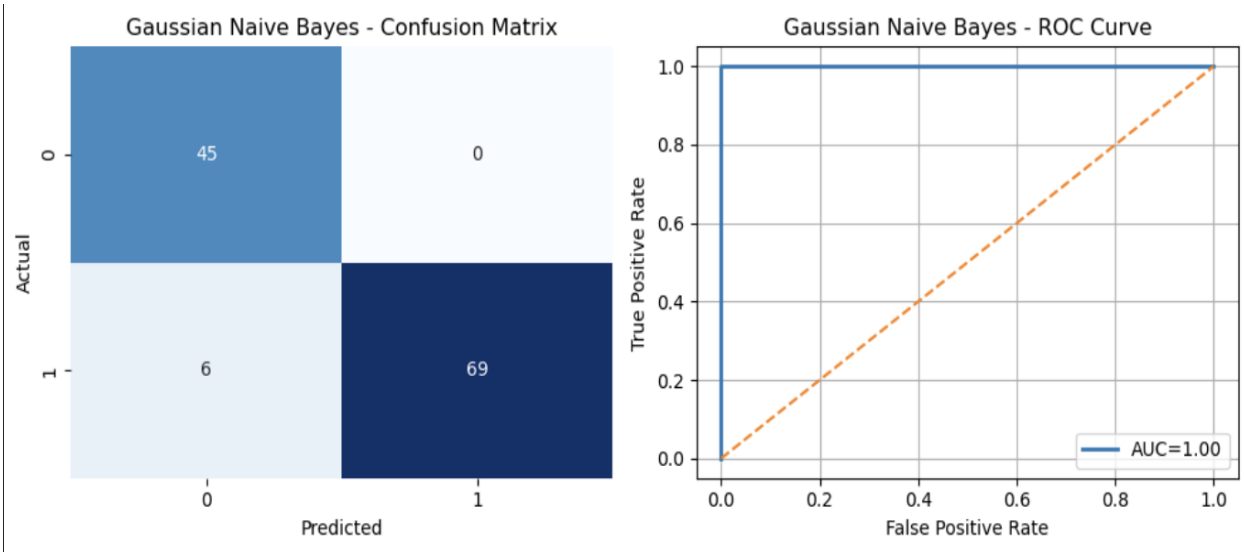
```
Best K-Nearest Params: {'metric': 'manhattan', 'n_neighbors': 11, 'weights': 'distance'}

==================== K-Nearest Neighbors ====================
Accuracy: 0.78
Confusion Matrix:
 [[42  3]
 [24 51]]
Classification report:
              precision    recall  f1-score   support

           0       0.64      0.93      0.76        45
           1       0.94      0.68      0.79        75

    accuracy                           0.78       120
   macro avg       0.79      0.81      0.77       120
weighted avg       0.83      0.78      0.78       120

AUC Score: 0.85
```



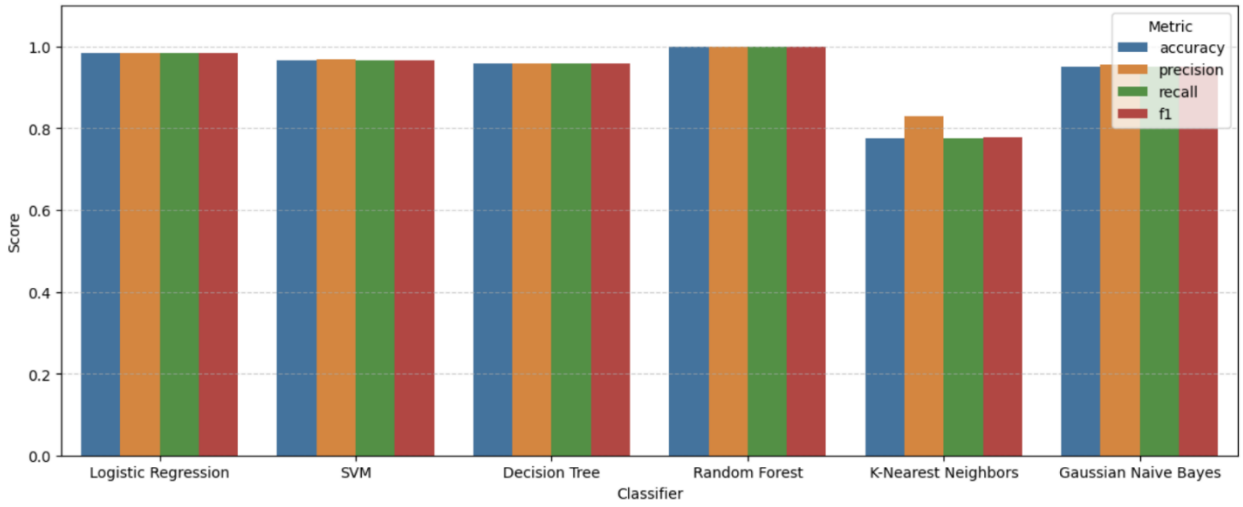


6. Gaussian Naïve Bayes:

```
==================== Gaussian Naive Bayes ====================
Accuracy: 0.95
Confusion Matrix:
 [[45  0]
 [ 6 69]]
Classification report:
              precision    recall  f1-score   support

           0       0.88      1.00      0.94        45
           1       1.00      0.92      0.96        75

    accuracy                           0.95       120
   macro avg       0.94      0.96      0.95       120
weighted avg       0.96      0.95      0.95       120

AUC Score: 1.00
```



Comparison across evaluation metrics:



Final Model Selection:

Chosen Model: Random Forest Classifier

Justification:

- After experimenting with multiple classification algorithms, the **Random Forest model** achieved the best performance on the test set, with the highest **accuracy, F1-score, and ROC-AUC**, and excellent **recall** for the CKD class.
- Considering the clinical importance of correctly identifying CKD patients and the robustness of tree-based ensembles, the **Random Forest classifier** was chosen as the final model for this problem.

Summary of Model Performance:

Model	Accuracy	Precision	Recall	F1 score
Logistic Regression	0.98	0.98	0.98	0.98
SVM	0.97	0.97	0.97	0.97
Decision Tree	0.96	0.96	0.96	0.96
Random Forest	1.00	1.00	1.00	1.00
K- Nearest Neighbors	0.78	0.83	0.78	0.78
Naïve Bayes	0.95	0.96	0.95	0.95