Assignment – Classification GridSearchCV Algorithms

Dataset File Name: **CKD.csv**

1. **Identify your problem statement:**

   3 – Stages:

   Stage -1: Domain → Machine Learning
   Stage -2: Learning → Supervised Learning
   Stage -3: Learning → Supervised Learning (Classification)

2. **Tell basic info about the dataset:**

   Total Number of rows: 399
   Total Number of Col: 28

3. **Mention the pre-processing method:**

   Get Dummies:

   String to Number – Ordinal data – Mapping – Label Encoder

Classification (Categorical)	Classification (NUMERICAL)
No	0
Yes	1

Standard Scalar:

```
from sklearn.preprocessing import StandardScaler
```

```
df4_num = df[['loan_amount','rate']]
df4_num.head()
```

client_id	loan_amount	rate
46109	13072	2.15
46109	9794	1.25
46109	12734	0.68
46109	12518	1.24
46109	14049	3.13

```
SS = StandardScaler()

scaled_x = SS.fit_transform(df4_num)
scaled_x
```

```
array([[ 1.36502962e+00, -4.45676908e-01],
       [ 4.34647423e-01, -8.21544457e-01],
       [ 1.13991132e+00, -1.05955306e+00],
       [ 1.08817014e+00, -8.25720763e-01],
       [ 1.45547678e+00, -3.63989115e-02],
       [-2.51263533e-01, -5.33379336e-01],
       [-4.33117505e-01,  2.61555546e+00],
       [ 1.12127812e+00,  3.85408004e-01],
       [ 7.38856969e-01,  9.45033020e-01],
       [-9.43412224e-01,  4.35523677e-01],
       [-1.52688038e+00,  1.36683993e+00],
       [ 1.01648330e-01, -1.13476741e+00],
       [ 2.57831622e-01, -9.84420394e-01],
```

4. Develop a good model with good evaluation metric:

- To find a good model for the dataset **CKD.csv** using the following machine learning classification algorithms:

1. Support Vector Classification
2. Decision Tree
3. Random Forest
4. Logistic Regression
5. K Nearest Neighbor
6. Naïve Bayes
7. Adaboost Classification
8. XG Boost Classification
9. LG Boost Classification

Support Vector Classification:

F1_Score: 0.9924946382275899

The confusion Matrix:

```
[[51  0]
 [ 1 81]]
```

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.98	1.00	0.99	51
1	1.00	0.99	0.99	82
accuracy			0.99	133
macro avg	0.99	0.99	0.99	133
weighted avg	0.99	0.99	0.99	133

roc_auc _score: 1.0

Decision Tree:

F1_Score: 0.9700283472213296

The confusion Matrix:

```
[[50  1]
 [ 3 79]]
```

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.94	0.98	0.96	51
1	0.99	0.96	0.98	82
accuracy			0.97	133
macro avg	0.97	0.97	0.97	133
weighted avg	0.97	0.97	0.97	133

roc_auc _score: 0.9719033955045432

Random Forest:

F1_Score: 0.9924667654735397

The confusion Matrix:

```
[[50  1]
 [ 0 82]]
```

Classification Report:

The report:

	precision	recall	f1-score	support
0	1.00	0.98	0.99	51
1	0.99	1.00	0.99	82
accuracy			0.99	133
macro avg	0.99	0.99	0.99	133
weighted avg	0.99	0.99	0.99	133

roc_auc _score: 0.9997608799617408

Logistic Regression:

F1_Score: 0.9924946382275899

The confusion Matrix:
[[51 0]
[1 81]]

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.98	1.00	0.99	51
1	1.00	0.99	0.99	82
accuracy			0.99	133
macro avg	0.99	0.99	0.99	133
weighted avg	0.99	0.99	0.99	133

roc_auc _score: 1.0

K-Nearest Neighbor:

F1_Score: 0.9701163285572423

The confusion Matrix:
[[51 0]
[4 78]]

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.93	1.00	0.96	51
1	1.00	0.95	0.97	82
accuracy			0.97	133
macro avg	0.96	0.98	0.97	133
weighted avg	0.97	0.97	0.97	133

roc_auc _score: 0.9873266379722621

Naïve Bayes (Gaussian) :

F1_Score: 0.9850141736106648

```
The confusion Matrix:
 [[51  0]
 [ 2 80]]
```

Classification Report:

```
The report:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        51
           1       1.00      0.98      0.99        82

    accuracy                           0.98       133
   macro avg       0.98      0.99      0.98       133
weighted avg       0.99      0.98      0.99       133
```

roc_auc _score: 1.0

AdaBoost Classification:

F1_Score: 0.9850141736106648

```
The confusion Matrix:
 [[51  0]
 [ 2 80]]
```

Classification Report:

```
The report:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        51
           1       1.00      0.98      0.99        82

    accuracy                           0.98       133
   macro avg       0.98      0.99      0.98       133
weighted avg       0.99      0.98      0.99       133
```

roc_auc _score: 1.0

XGBoost Classification:

F1_Score: 0.47029200909249863

```
The confusion Matrix:
 [[ 0 51]
 [ 0 82]]
```

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	51
1	0.62	1.00	0.76	82
accuracy			0.62	133
macro avg	0.31	0.50	0.38	133
weighted avg	0.38	0.62	0.47	133

roc_auc _score: 0.9817073170731707

LGBoost Classification:

F1_Score: 0.47029200909249863

```
The confusion Matrix:
 [[ 0 51]
 [ 0 82]]
```

Classification Report:

The report:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	51
1	0.62	1.00	0.76	82
accuracy			0.62	133
macro avg	0.31	0.50	0.38	133
weighted avg	0.38	0.62	0.47	133

roc_auc _score: 0.9966523194643712

5. **All the research values of each algorithm have been documented.**

6. **Mention your final model, justify why u have chosen the same:**

To use the above algorithms for this dataset, we have finally concluded and found the best model for the **CKD.csv** dataset. There are two best models that worked perfectly for this dataset and provided the best results. All the validating parameters for the classification algorithms, such as F1 score, confusion matrix, classification report, and ROC_AUC score, are equal when compared for both **Support Vector Classification** and **Logistic Regression** algorithms. These two classification algorithms yielded better results, especially with accuracy, precision, and recall at 0.99, while the ROC_AUC score is 1.0. Therefore, we can use these two algorithms and proceed to the next step of the deployment phase.