Machine Learning - Classification Assignment

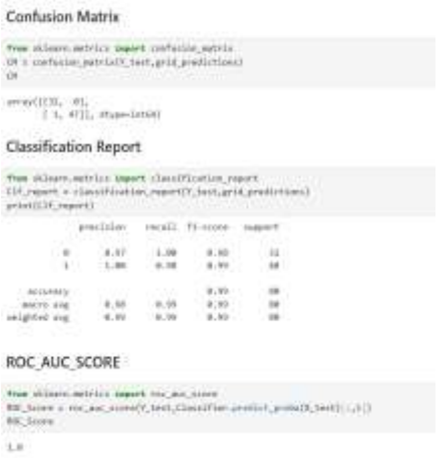
1. Identify your problem statement

Goal	:	CHRONIC KIDNEY DISEASE PREDICTION
3 Stages	:	Machine Learning (number based data)
		Supervised Learning (Input & Output present)
		Classification Algorithms Used - Logistic Resgression, K Nearest Neighbors, Naive Bayes(Gaussian, Bernoulli, Multinomial), Support Vector Machince, Decision Tree & Random Forest

2. No of rows & columns : 399 rows   25 columns

3. Pre-processing method : Categorical data - Nominal - One Hot Encoding Method (pandas function - get_dummies)

Support Vector Machine :



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [ 1, 47]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

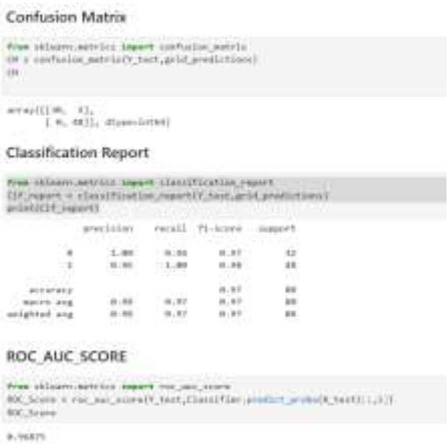
```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        32
           1       1.00      0.98      0.99        48

    accuracy                           0.99        80
   macro avg       0.98      0.99      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
1.0
```

Decision Tree :



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[30,  2],
       [ 0, 48]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

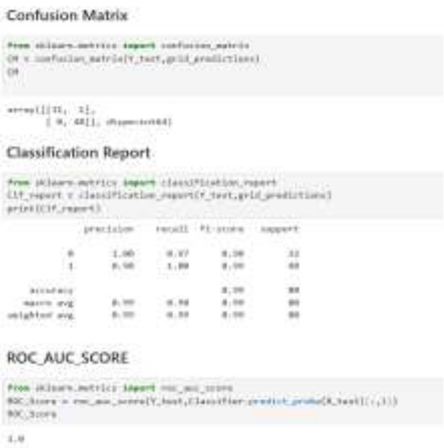
```
              precision    recall  f1-score   support

           0       1.00      0.94      0.97        32
           1       0.96      1.00      0.98        48

    accuracy                           0.97        80
   macro avg       0.98      0.97      0.97        80
weighted avg       0.98      0.97      0.97        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
0.96875
```

Random Forest:



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[31,  1],
       [ 0, 48]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

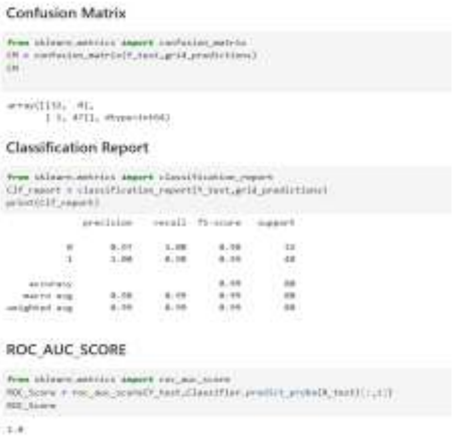
```
              precision    recall  f1-score   support

           0       1.00      0.97      0.98        32
           1       0.98      1.00      0.99        48

    accuracy                           0.99        80
   macro avg       0.99      0.98      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
1.0
```

Logistic Regression



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [ 1, 47]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

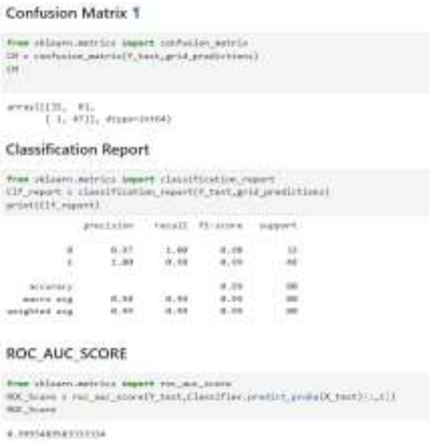
```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        32
           1       1.00      0.98      0.99        48

    accuracy                           0.99        80
   macro avg       0.98      0.99      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
1.0
```

KNN :



Confusion Matrix 1

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [ 1, 47]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

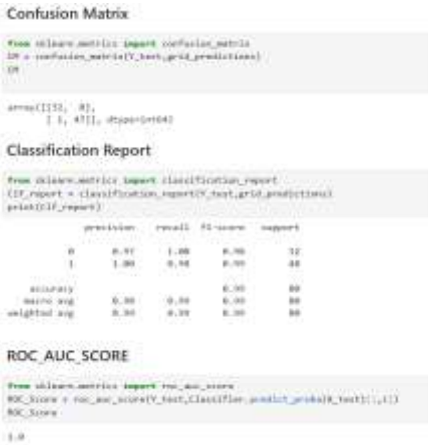
```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        32
           1       1.00      0.98      0.99        48

    accuracy                           0.99        80
   macro avg       0.98      0.99      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
0.9954861111111114
```

Naive Bayes : 1. Guassian NB



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [ 1, 47]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

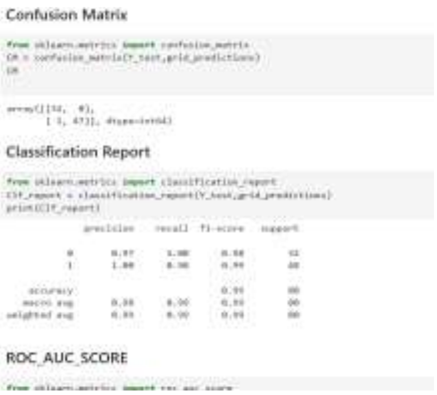
```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        32
           1       1.00      0.98      0.99        48

    accuracy                           0.99        80
   macro avg       0.98      0.99      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
1.0
```

Naive Bayes : 2. Bernoulli NB



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [ 1, 47]], dtype=int64)
```

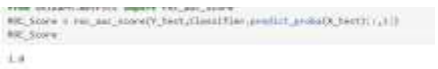
Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        32
           1       1.00      0.98      0.99        48

    accuracy                           0.99        80
   macro avg       0.98      0.99      0.99        80
weighted avg       0.99      0.99      0.99        80
```

ROC_AUC_SCORE

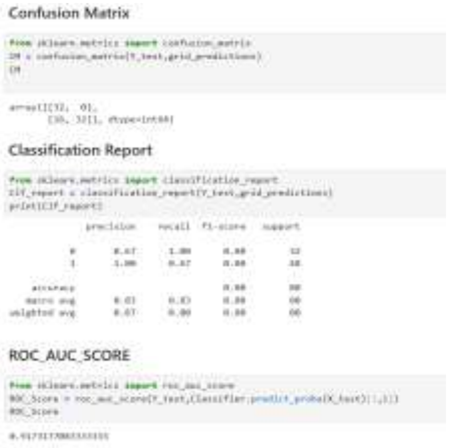
```
from sklearn.metrics import roc_auc_score
```



```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
1.0
```

Naive Bayes : 2. Multinomial NB



Confusion Matrix

```
from sklearn.metrics import confusion_matrix
CM = confusion_matrix(Y_test,grid_predictions)
CM
```

```
array([[32,  0],
       [18, 32]], dtype=int64)
```

Classification Report

```
from sklearn.metrics import classification_report
Clf_report = classification_report(Y_test,grid_predictions)
print(Clf_report)
```

```
              precision    recall  f1-score   support

           0       0.67      1.00      0.80        32
           1       1.00      0.47      0.64        34

    accuracy                           0.80        66
   macro avg       0.83      0.83      0.80        66
weighted avg       0.87      0.80      0.80        66
```

ROC_AUC_SCORE

```
from sklearn.metrics import roc_auc_score
ROC_Score = roc_auc_score(Y_test,Classifier.predict_proba(X_test)[:,1])
ROC_Score
```

```
0.9173177083333333
```

Result :

1. **For this given dataset "Random Forest Classifier Algorithm" predicts the better model (99% Accuracy) compared to others.**
2. **Accuracy, Recall, Precision, F1 Score, Macro Average & Weighted Average values of all the algorithms mentioned above.**
3. **ROC_AUC_SCORE : Here SVM, Random Forest, Logistic Regression, Gaussian & Bernoulli results in better classification Perfomance.**