CKD CLASSIFICATION ASSIGNMENT

Problem Statement: Predicting Chronic Kidney Disease

*Logistic Regression

```
grid_prediction=grid.predict(x_test)
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,grid_prediction)
print(cm)
```

```
[[45  0]
 [ 1 74]]
```

```
from sklearn.metrics import classification_report
clf_report=classification_report(y_test,grid_prediction)
print(clf_report)
```

```
              precision    recall  f1-score   support

       False       0.98      1.00      0.99        45
        True       1.00      0.99      0.99        75

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120
```

***Decision Tree:**

```
[9]: grid_prediction=grid.predict(x_test)
     from sklearn.metrics import confusion_matrix
     cm=confusion_matrix(y_test,grid_prediction)
     print(cm)

     [[45  0]
      [ 3 72]]

[10]: from sklearn.metrics import classification_report
      clf_report=classification_report(y_test,grid_prediction)
      print(clf_report)

                   precision    recall  f1-score   support

            False       0.94      1.00      0.97        45
             True       1.00      0.96      0.98        75

         accuracy                           0.97       120
        macro avg       0.97      0.98      0.97       120
     weighted avg       0.98      0.97      0.98       120
```

***SVM:**

```
[10]: grid_prediction=grid.predict(x_test)
      from sklearn.metrics import confusion_matrix
      cm=confusion_matrix(y_test,grid_prediction)
      print(cm)

      [[45  0]
       [ 2 73]]

[11]: from sklearn.metrics import classification_report
      clf_report=classification_report(y_test,grid_prediction)
      print(clf_report)

                   precision    recall  f1-score   support

            False       0.96      1.00      0.98        45
             True       1.00      0.97      0.99        75

         accuracy                           0.98       120
        macro avg       0.98      0.99      0.98       120
     weighted avg       0.98      0.98      0.98       120
```

***Random Forest:**

```
[15]: grid_prediction=grid.predict(x_test)
      from sklearn.metrics import confusion_matrix
      cm=confusion_matrix(y_test,grid_prediction)
      print(cm)

      [[45  0]
       [ 1 74]]

[16]: from sklearn.metrics import classification_report
      clf_report=classification_report(y_test,grid_prediction)
      print(clf_report)

                    precision    recall  f1-score   support

             False       0.98      1.00      0.99        45
              True       1.00      0.99      0.99        75

          accuracy                           0.99       120
         macro avg       0.99      0.99      0.99       120
      weighted avg       0.99      0.99      0.99       120
```

***KNearestNeighbor:**

```
[8]: grid_prediction=grid.predict(x_test)
     from sklearn.metrics import confusion_matrix
     cm=confusion_matrix(y_test,grid_prediction)
     print(cm)

     [[45  0]
      [ 6 69]]

[9]: from sklearn.metrics import classification_report
     clf_report=classification_report(y_test,grid_prediction)
     print(clf_report)

                   precision    recall  f1-score   support

            False       0.88      1.00      0.94        45
             True       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
```

Best Model:

Saving Logistic Regression & Random Forest as best model.

PreProcessing Technique:

Standard Scaler used

Evaluation Metrics:

Roc Auc used

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

```
auc_score = roc_auc_score(y_test,grid_prediction)
print(f"ROC AUC Score: {auc_score:.2f}")
```

```
ROC AUC Score: 0.99
```