Chronic kidney Disease Dataset

1. Identify your problem statement
    - Machine Learning➔ Supervised learning ➔Classification
2. Tell basic info about the dataset (Total number of rows, columns)
    - The client requirement is Chronic Kidney Disease (CKD) predict.
    - Rows 399 columns 24+1 class
3. Mention the pre-processing method if you're doing any (like converting string to number – nominal data)
    - This dataset is not for ordering .this dataset is nominal data
    - Encode the labels for the nominal attribute values
    - Use the Standard Scale
4. Develop a good model with good evaluation metric. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final mode.
    - Decision Tree Classifier object
    - Random Forest object
    - SVM classified object
    - Gaussian Naive Bayes object
    - K-Neighbors Classifier object
    - Logistic Regression object
    - Finally the best model for the SVM Good model.
5. All the research values of each algorithm should be documented. (You can make tabulation or screenshot of the results.)

Decision Tree

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits
the confusion Matrix:
 [[51  0]
 [ 5 77]]
The is DecisionTree:
              precision    recall  f1-score   support

           0       0.91      1.00      0.95        51
           1       1.00      0.94      0.97        82

    accuracy                           0.96       133
   macro avg       0.96      0.97      0.96       133
weighted avg       0.97      0.96      0.96       133
```

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

```
0.9695121951219512
```

Random Forest

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits
C:\Users\DELL\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:909: DataConversionWarnin
expected. Please change the shape of y to (n_samples,), for example using ravel().
  self.best_estimator_.fit(X, y, **fit_params)
the confusion Matrix:
 [[51  0]
 [ 1 81]]
The is RandomForest:
              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
```

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

```
0.9997608799617408
```

SVM classified

```
Fitting 5 folds for each of 32 candidates, totalling 160 fits
the confusion Matrix:
 [[51  0]
 [ 1 81]]
The is SVM:
              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
```

```
C:\Users\DELL\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarni
ng: A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Gaussian Naive Bayes

```
the confusion Matrix:
 [[51  0]
 [ 3 79]]
The is Gaussian Naive Bayes:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97        51
           1       1.00      0.96      0.98        82

    accuracy                           0.98       133
   macro avg       0.97      0.98      0.98       133
weighted avg       0.98      0.98      0.98       133
```

```
C:\Users\DELL\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

8]:
```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

8]:
```
1.0
```

K-Neighbors Classifier

```
C:\Users\DELL\anaconda3\Lib\site-packages\sklearn\neighbors\_classification.py:215: DataConve
as expected. Please change the shape of y to (n_samples,), for example using ravel().
  return self._fit(X, y)
the confusion Matrix:
 [[51  0]
 [ 4 78]]
The is KNeighbors:
              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
```

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

```
1.0
```

Logistic Regression object

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
The confusion Matrix:
 [[51  0]
 [ 1 81]]
The is LogisticRegression:
              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
```

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

   - Accuracy was fond to be higher for svm Random Forest classifier, Logistic regression.
   - Comparatively svm class another class accuracy low value