Assignment - Problem Identification of Chronic Kidney Disease (CKD).

1. Problem Statement

Dataset contains Patient's Details with those details we have to create a model and predict the patient who will be affected by Kidney Disease in future.

Stage 1-> Machine Learning

Stage2 -> Supervised Learning

Stage 3- > Classification

2. Total No of Rows 399, Columns 25.
24 /p columns , 1 o/p column.

3. Converted Categorical data into Nominal data after converting dataset contains 399 rows and 28 columns, which means 27 inputs and 1 output. Here we are using a pre-processing technique is called one hot encoding
4. Using this dataset, we developed the following good models to good evaluation metrics and Finally we have to come up with final model.
    - SVM classifier
    - Decision Tree Classifier
    - Random Forest Classifer
    - Logistic Regression
    - KNN Classifier

5. All the Research value of each Algorithms are listed below:

SVM Classifier.

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,y_pred,average='weighted')
print("The f1_macro value for best parameter{}".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'C': 10, 'gamma': 'auto', 'kernel': 'sigmoid'} 0.9850141736106648
```

```
print(cm)
```

```
[[51  0]
 [ 2 80]]
```

```
print(clf_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
```

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

```
1.0
```

Decision Tree Classifier.

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,y_pred,average='weighted')
print("The f1_macro value for best parameter{}".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'} 0.9924946382275899
```

```
print(cm)
```

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

```
print(clf_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
```

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

```
0.9939024390243902
```

RandomForest Classifier

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,y_pred,average='weighted')
print("The f1_macro value for best parameter{}".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'criterion': 'entropy', 'max_features': 'log2', 'n_estimators': 100} 0.9849624060150376
```

```
print(cm)
```

```
[[50  1]
 [ 1 81]]
```

```
print(clf_report)
```

```
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        51
           1       0.99      0.99      0.99        82

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

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

```
0.9997608799617408
```

Logistic Regression.

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,y_pred,average='weighted')
print("The f1_macro value for best parameter{}".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'penalty': 'l2', 'solver': 'newton-cg'} 0.9924946382275899
```

```
print(cm)
```

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

```
print(clf_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
```

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

```
1.0
```

KNN Classifier

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,y_pred,average='weighted')
print("The f1_macro value for best parameter{}".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'metric': 'minkowski', 'n_neighbors': 5, 'p': 1, 'weights': 'uniform'} 0.9626932787797391
```

```
print(cm)
```

```
[[51  0]
 [ 5 77]]
```

```
print(clf_report)
```

```
              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])
```

```
0.9934241989478718
```

6. Finally we concluded Our Final Good model as SVM Classifier. Because, its f1_score is 0.99 and rou_auc_score is 1.0 which values are best and higher than the other models. Logistic Regression also the Best Model , We choosen the SVM Classifier is the Good Model when we compared the values of f1_ score, rou_auc_score values