Week – 4

Classification Assignment

Problem Statement:

A The problem statement revolves around leveraging machine learning techniques to develop a predictive model for Chronic Kidney Disease (CKD). Chronic Kidney Disease is a serious medical condition characterized by the gradual loss of kidney function over time. Early detection and prediction of CKD can be crucial for timely medical intervention and better patient outcomes.

Tell basic info about the dataset:

Total number of rows: 399
Total number of columns: 25
Null values: Nil
Affected by CKD: 249
Not affected by CKD: 150

Mention the pre-processing method:

- **Converting Categorical to Numeric**: Categorical variables like 'rbc,' 'pc,' 'pcc,' 'ba,' 'htn,' 'dm,' 'cad,' 'appet,' 'pe,' 'ane' need to be converted to numeric format using techniques like one-hot encoding or label encoding.

- **Scaling Numerical Features**: Numerical features may need to be scaled to bring them to a similar scale.

Develop a good model with good evaluation metric:

Logistic Regression

In [16]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter {'multi_class': 'multinomial', 'penalty': 'l2', 'solver': 'newton-cg'}: 0.992
4946382275899
```

In [17]:
```
print("The confusion Matrix:\n",cm)
```

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

In [18]:
```
print("The report:\n",clf_report)
```

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

       False       0.98      1.00      0.99        51
        True       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
```

In [20]:
```
from sklearn.metrics import roc_auc_score

roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
```

Out[20]:
```
1.0
```

Decision Tree Classifier

In [16]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,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'}: 0.962
5928174473452
```

In [17]:
```
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[50  1]
 [ 4 78]]
```

In [18]:
```
print("The report:\n",clf_report)
```

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

           0       0.93      0.98      0.95        51
           1       0.99      0.95      0.97        82

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

In [19]:
```
from sklearn.metrics import roc_auc_score

roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
```

Out[19]:
```
0.9658058345289334
```

Support Vector Machine

In [16]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,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'}: 0.9924946382275899
```

In [17]:
```
print("The confusion Matrix:\n",cm)
```

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

In [18]:
```
print("The report:\n",clf_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
```

In [19]:
```
from sklearn.metrics import roc_auc_score

roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
```

Out[19]:
```
1.0
```

Random Forest Classifier

In [16]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter {'criterion': 'gini', 'max_features': 'log2', 'n_estimators': 10}: 0.99249463
82275899
```

In [17]:
```
print("The confusion Matrix:\n",cm)
```

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

In [18]:
```
print("The report:\n",clf_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
```

In [19]:
```
from sklearn.metrics import roc_auc_score

roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
```

Out[19]:
```
0.9998804399808705
```

Mention your final model, justify why?

The Random Forest model is selected as the final model for predicting Chronic Kidney Disease. Here's a justification for this choice:

- **Ensemble Nature**: Random Forest is an ensemble of decision trees, which helps mitigate overfitting and improves generalization to new data.
- **Balanced Performance**: Random Forest tends to offer a good balance between interpretability and predictive performance.
- **Robustness**: Random Forest is generally robust to outliers and noise in the data.
- **Generalization**: The ensemble nature of Random Forest makes it more likely to generalize well to new, unseen data