Project Development Phase
Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID27851
Project Name	Early Detection Of Chronic Kidney Disease Using Machine Learning
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model:** Confusion Matrix – [[68, 2], [1,29]] Accuracy Score- Training accuracy score : 1.0 Testing accuracy score : 0.97 Classification Report – 0.97 , 0.97 , 0.97 , 100	
2.	Tune the Model	Hyperparameter Tuning – GridSearchCV Validation Method – Training accuracy score : 1.0 Testing accuracy score : 0.97	

1 . METRICS :

CLASSIFICATION MODEL:

CONFUSION MATRIX :

CONFUSION MATRIX

```
In [118]: from sklearn.metrics import confusion_matrix
          confusion_matrix(y_test,test_pred)

Out[118]: array([[68,  2],
                 [ 1, 29]], dtype=int64)
```

```
In [119]: pd.crosstab(y_test,test_pred)

Out[119]:
```

col_0	0	1
classification		
0	68	2
1	1	29

ACCURACY SCORE :

ACCURACY SCORE

```
In [95]: from sklearn.metrics import accuracy_score
         print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred))
         print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred))

TRAINING ACCURACY :  1.0
TESTING ACCURACY :  0.97
```

CLASSIFICATION REPORT :

```
In [97]: from sklearn.metrics import classification_report
         classification_report(y_test,test_pred)

Out[97]: '              precision    recall  f1-score   support\n\n           0       0.99      0.97      0.98        70\n           1
         0.94      0.97      0.95        30\n\n    accuracy                           0.97       100\n   macro avg       0.96      0.97
         0.96       100\nweighted avg       0.97      0.97      0.97       100\n'
```

2 . TUNE THE MODEL :

HYPER PARAMETER TUNING :

HYPER PARAMETER TUNING

In [99]:
```
parameters = {"criterion":['gini','entropy'],"max_depth":range(1,10),"random_state":[50,60]}
```

In [100]:
```
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(model1, parameters,cv = 10,verbose = 1)
grid.fit(X_train,y_train)
```

```
Fitting 10 folds for each of 36 candidates, totalling 360 fits
```

Out[100]:
```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': range(1, 10), 'random_state': [50, 60]},
             verbose=1)
```

In [101]:
```
grid.best_params_
```

Out[101]:
```
{'criterion': 'gini', 'max_depth': 6, 'random_state': 60}
```

In [102]:
```
grid.best_estimator_
```

Out[102]:
```
DecisionTreeClassifier(max_depth=6, random_state=60)
```

In [103]:
```
grid.best_score_
```

Out[103]:
```
0.9833333333333334
```

In [104]:
```
model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
```

VALIDATION :

In [104]:
```
model2 = DecisionTreeClassifier(max_depth = 6,criterion = 'gini',random_state = 60)
```

In [105]:
```
model2.fit(X_train,y_train)
```

Out[105]:
```
DecisionTreeClassifier(max_depth=6, random_state=60)
```

In [106]:
```
train_pred2 = model1.predict(X_train)
test_pred2 = model1.predict(X_test)
```

In [107]:
```
print('TRAINING ACCURACY : ',accuracy_score(y_train,train_pred2))
print('TESTING ACCURACY : ',accuracy_score(y_test,test_pred2))
```

```
TRAINING ACCURACY :  1.0
TESTING ACCURACY :  0.97
```

In [108]:
```
pd.crosstab(y_test,test_pred2)
```

Out[108]:

col_0	0	1
classification		
0	68	2
1	1	29