Project Development Phase
Model Performance Test

Team-ID-592870

Model Performance Testing:

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

S. No.	Parameter	Values
1.	Metrics	**Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report -
2.	Tune the Model	Hyperparameter Tuning

Classification Model:

Confusion Matrix - , Accuray Score- & Classification Report -

In [35]:
```
dtc=DecisionTreeClassifier()
dtc.fit(X_train,y_train)

y_test_predict2=dtc.predict(X_test)
test_accuracy=accuracy_score(y_test,y_test_predict2)
test_accuracy
```

Out[35]: 0.988

In [36]:
```
y_train_predict2=dtc.predict(X_train)
train_accuracy=accuracy_score(y_train,y_train_predict2)
train_accuracy
```

Out[36]: 1.0

In [39]:
```
pd.crosstab(y_test,y_test_predict2)
```

Out[39]:

col_0	Fraud	No Fraud
isFraud		
Fraud	1	2
No Fraud	4	493

In [40]:
```
print(classification_report(y_test,y_test_predict2))
```

```
              precision    recall  f1-score   support

       Fraud       0.20      0.33      0.25         3
    No Fraud       1.00      0.99      0.99       497

    accuracy                           0.99       500
   macro avg       0.60      0.66      0.62       500
weighted avg       0.99      0.99      0.99       500
```

Confusion Matrix

In [51]:
```
cm = confusion_matrix(y_test, y_test_predict2)
print("Confusion Matrix:")
print(cm)
```

```
Confusion Matrix:
[[  1   2]
 [  4 493]]
```

Hyperparameter Tuning

In [52]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier

# Define the hyperparameters and their possible values for tuning
param_grid = {
    'n_neighbors': [3, 5, 7, 9],
    'metric': ['euclidean', 'manhattan', 'minkowski']
}

# Create the KNeighborsClassifier model
k_classifier = KNeighborsClassifier()

# Perform Grid Search with cross-validation
grid_search = GridSearchCV(estimator=k_classifier, param_grid=param_grid, scoring='accuracy', cv=5)
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_params = grid_search.best_params_
print("Best Hyperparameters:", best_params)
```

```
Best Hyperparameters: {'metric': 'euclidean', 'n_neighbors': 3}
```