Project Development Phase
Model Performance Test

Date	22 NOvember 2023
Team ID	Team-591594
Project Name	Project -Online Payments Fraud Detection using ML
Maximum Marks	10 Marks

Model Performance Testing:

Metrics

Regression metrics:

```python
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score, confusion_matrix, accuracy_score, classification_report
# Calculate regression metrics
mae = mean_absolute_error(y_test, pred)
mse = mean_squared_error(y_test, pred)
rmse = mean_squared_error(y_test, pred, squared=False)
r2 = r2_score(y_test,pred)

print("Regression Metrics:")
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r2)
```

```
Regression Metrics:
MAE: 0.0011921189698583289
MSE: 0.0011921189698583289
RMSE: 0.034527075894988976
R2 Score: 0.07437002241499024
```

Classification Metrics:

Confusion Matrix

```python
[161] confusion_matrix(y_test,pred)
```

```
array([[1270880,       3],
       [   1514,     127]])
```

Accuray Score

```python
[160] print(accuracy_score(y_test,pred))
```

```
0.9988078810301416
```

Classification Report

```
[69] print(classification_report(y_test,pred))

              precision    recall  f1-score   support

           0       1.00      1.00      1.00   1270883
           1       0.98      0.08      0.14      1641

    accuracy                           1.00   1272524
   macro avg       0.99      0.54      0.57   1272524
weighted avg       1.00      1.00      1.00   1272524
```

Tune the Model:

Hyperparameter Tuning

Hyperparameter tuning

```
from sklearn.model_selection import GridSearchCV
# 3. Hyperparameter Tuning
# Define hyperparameters to tune
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100]}

# Perform grid search with cross-validation
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')
grid_search.fit(x_train, y_train)

# Print the best hyperparameters
best_params = grid_search.best_params_
print("\nBest Hyperparameters:", best_params)

# Print the best model's accuracy on the test set
best_model = grid_search.best_estimator_
best_model_accuracy = best_model.score(x_test, y_test)
print("Best Model Accuracy on Test Set:", best_model_accuracy)
```

```
Best Hyperparameters: {'C': 100}
Best Model Accuracy on Test Set: 0.9991080718320441
```

Validation Method

Validation Method

```python
# Import necessary libraries
from sklearn.model_selection import cross_val_score, KFold
from sklearn.linear_model import LogisticRegression

# Assuming you have a DataFrame 'X' containing features and a Series 'y' containing target values

# 1. Regression Task: Cross-validation for R2 score
# Specify the number of folds (e.g., 5-fold cross-validation)
cv_r2 = cross_val_score(model, X_scaled, y, cv=5, scoring='r2')

# Print the cross-validated R2 scores
print("Cross-validated R2 scores:", cv_r2)
print("Mean R2 score:", cv_r2.mean())

# 2. Classification Task: Cross-validation for Accuracy
# Assume binary classification (modify accordingly for multi-class)
cv_accuracy = cross_val_score(model, X_scaled, y, cv=5, scoring='accuracy')

# Print the cross-validated accuracy scores
print("\nCross-validated Accuracy scores:", cv_accuracy)
print("Mean Accuracy:", cv_accuracy.mean())
```

```
Cross-validated R2 scores: [ 0.04993121  0.06639581  0.07732361 -0.05553204  0.09804422]
Mean R2 score: 0.04723256211112801

Cross-validated Accuracy scores: [0.99877566 0.99879688 0.99881024 0.99863893 0.99883696]
Mean Accuracy: 0.9987717323995462
```