Project Development Phase
Model Performance Test

Date	09 November 2023
Project Name	Project - Understanding Audience: A Machine Learning Approach to Customer Segmentation
Maximum Marks	10 Marks

Model Performance Testing:

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

S.No.	Parameter	Values
1.	Metrics	**Regression Model:** MAE -, MSE -, RMSE-, R2 score - **Classification Model:** Confusion Matrix -, Accuracy Score- & Classification Report -
2.	Tune the Model	Hyperparameter Tuning - Validation Method -

METRICS:

Regression Model:

MAE:

```
from sklearn.metrics import mean_absolute_error

print ('MAE =',mean_absolute_error(y_test, y_pred))
```

```
MAE = 0.0
```

MSE:

```
from sklearn.metrics import mean_squared_error

print ('MSE =',mean_squared_error(y_test,y_pred ))
```

```
MSE = 0.0
```

RMSE:

```
from sklearn.metrics import mean_squared_error
import math

RMSE = math.sqrt(MSE)
print("Root Mean Square Error:\n")
print(RMSE)
```

```
Root Mean Square Error:

0.0
```

R2 Score:

```
from sklearn.metrics import r2_score

print ('R Squared =',r2_score(y_test,y_pred ))
```

```
R Squared = 1.0
```

Classification Model:

Confusion Matrix:

```
confusion = confusion_matrix(y_test, y_pred)
confusion
```

```
array([[ 65,   0,   0],
       [  0, 275,   0],
       [  0,   0, 260]])
```

Accuracy Score:

```
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

```
1.0
```

Classification Report:

```
classification_rep = classification_report(y_test, y_pred)
classification_rep
```

```
'              precision    recall  f1-score   support\n\n           0       1.00      1.00      1.00        65\n           1       1.00      1.00      1.00
1.00       275\n           2       1.00      1.00      1.00       260\n\n    accuracy                           1.00       600\n   macro avg       1.0
0      1.00      1.00       600\nweighted avg       1.00      1.00      1.00       600\n'
```

TUNE THE MODEL:

Hyperparameter Tuning:

```
grid_search = GridSearchCV(model, param_grid, cv=5)

grid_search.fit(x, y)

best_params = grid_search.best_params_
best_score = grid_search.best_score_

print("Best Parameters:", best_params)
print("Best Score:", best_score)
```

```
Best Parameters: {'max_depth': 20, 'min_samples_split': 10, 'n_estimators': 100}
Best Score: 0.9984999999999999
```

Validation Method:

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

k = 5
kf = KFold(n_splits=k, shuffle=True, random_state=42)

cv_scores = cross_val_score(model, x, y, cv=kf)

print(f'Cross-Validation Scores: {cv_scores}')
print(f'Mean CV Score: {cv_scores.mean()}')
```

```
Cross-Validation Scores: [1.     1.     0.9975 0.9975 1.    ]
Mean CV Score: 0.999
```