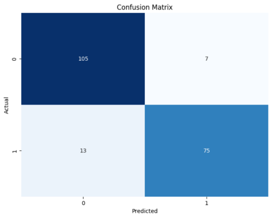
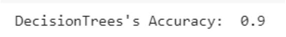
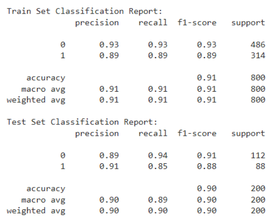
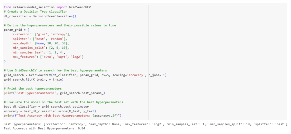
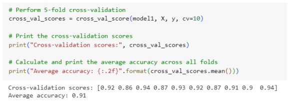
Project Development Phase
Model Performance Test

Date	10 November 2023
Team ID	EXT2023TMID591615
Project Name	Car Purchase Prediction Model
Maximum	15 Marks

Model Performance Testing:

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

S.No	Parameter	Values	Screenshot
1.	Metrics	**Classification Model: Decision Tree** Confusion Matrix Accuracy Score Classification Report	  
2.	Tune The Model	Hyperparameter Tuning: Grid Search CV Validation Method: 5 fold cross validation.	 

```
DecisionTrees's Accuracy:  0.9
```

```
Train Set Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.93      0.93       486
           1       0.89      0.89      0.89       314

    accuracy                           0.91       800
   macro avg       0.91      0.91      0.91       800
weighted avg       0.91      0.91      0.91       800

Test Set Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.94      0.91       112
           1       0.91      0.85      0.88        88

    accuracy                           0.90       200
   macro avg       0.90      0.89      0.90       200
weighted avg       0.90      0.90      0.90       200
```

```
from sklearn.model_selection import GridSearchCV
# Create a Decision Tree classifier
dt_classifier = DecisionTreeClassifier()

# Define the hyperparameters and their possible values to tune
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt', 'log2']
}

# Use GridSearchCV to search for the best hyperparameters
grid_search = GridSearchCV(dt_classifier, param_grid, cv=5, scoring='accuracy', n_jobs=-1)
grid_search.fit(X_train, y_train)

# Print the best hyperparameters
print("Best Hyperparameters:", grid_search.best_params_)

# Evaluate the model on the test set with the best hyperparameters
best_dt_classifier = grid_search.best_estimator_
accuracy = best_dt_classifier.score(X_test, y_test)
print(f"Test Accuracy with Best Hyperparameters: {accuracy:.2f}")
```

```
Best Hyperparameters: {'criterion': 'entropy', 'max_depth': None, 'max_features': 'log2', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'best'}
Test Accuracy with Best Hyperparameters: 0.84
```

```
# Perform 5-fold cross-validation
cross_val_scores = cross_val_score(model1, X, y, cv=10)

# Print the cross-validation scores
print("Cross-validation scores:", cross_val_scores)

# Calculate and print the average accuracy across all folds
print("Average accuracy: {:.2f}".format(cross_val_scores.mean()))
```

```
Cross-validation scores: [0.92 0.86 0.94 0.87 0.93 0.92 0.87 0.91 0.9  0.94]
Average accuracy: 0.91
```