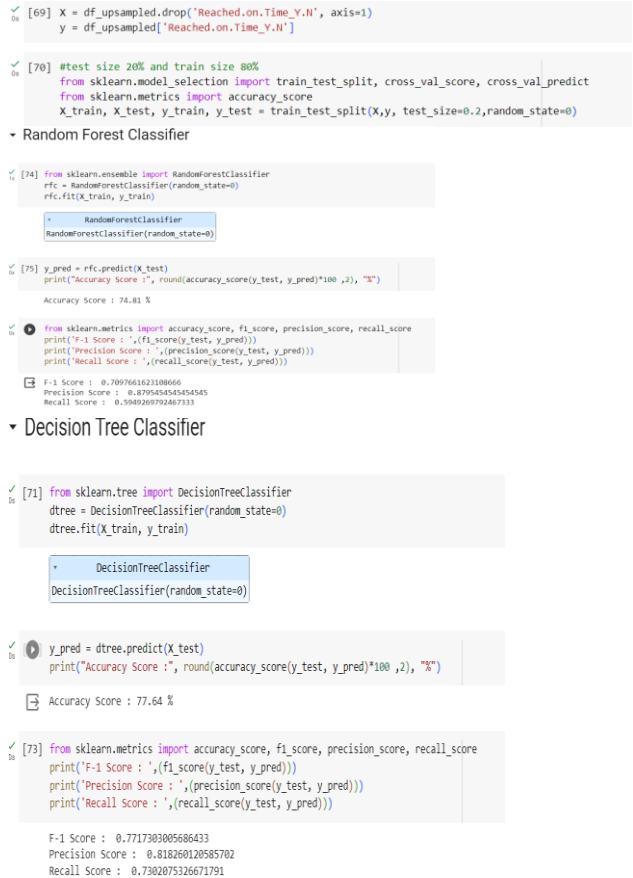


Project Development Phase Model Performance Test

Date	22 November 2023
Team ID	Team- 591756
Project Name	ECOMMERCE SHIPPING PREDICTION 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	<p>Regression Model: MAE - , MSE - , RMSE - , R2 score -</p> <p>Classification Model: Confusion Matrix - , Accuracy Score- & Classification Report -</p>	 <pre> [69] X = df_upsampled.drop("Reached.on.Time_Y.N", axis=1) y = df_upsampled["Reached.on.Time_Y.N"] [70] #test size 20% and train size 80% from sklearn.model_selection import train_test_split, cross_val_score, cross_val_predict from sklearn.metrics import accuracy_score X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=0) • Random Forest Classifier [74] from sklearn.ensemble import RandomForestClassifier rfc = RandomForestClassifier(random_state=0) rfc.fit(X_train, y_train) + RandomForestClassifier RandomForestClassifier(random_state=0) [75] y_pred = rfc.predict(X_test) print("Accuracy Score :", round(accuracy_score(y_test, y_pred)*100 ,2), "%") Accuracy Score : 74.81 % [76] from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score print("F-1 Score : ",(f1_score(y_test, y_pred))) print("Precision Score : ",(precision_score(y_test, y_pred))) print("Recall Score : ",(recall_score(y_test, y_pred))) F-1 Score : 0.7097061623108666 Precision Score : 0.8795454545454545 Recall Score : 0.5949269792467333 • Decision Tree Classifier [71] from sklearn.tree import DecisionTreeClassifier dtree = DecisionTreeClassifier(random_state=0) dtree.fit(X_train, y_train) + DecisionTreeClassifier DecisionTreeClassifier(random_state=0) [72] y_pred = dtree.predict(X_test) print("Accuracy Score :", round(accuracy_score(y_test, y_pred)*100 ,2), "%") Accuracy Score : 77.64 % [73] from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score print("F-1 Score : ",(f1_score(y_test, y_pred))) print("Precision Score : ",(precision_score(y_test, y_pred))) print("Recall Score : ",(recall_score(y_test, y_pred))) F-1 Score : 0.7717303005686433 Precision Score : 0.818260120585702 Recall Score : 0.7302075326671791 </pre>

			<div> <div>Logistic Regression</div> <pre> [77] from sklearn.linear_model import LogisticRegression lr = LogisticRegression(random_state=0) lr.fit(X_train, y_train) LogisticRegression LogisticRegression(random_state=0) [78] y_pred = lr.predict(X_test) print('Accuracy Score : ', round(accuracy_score(y_test, y_pred)*100, 2), "%") Accuracy Score : 68.52 % from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score print('F-1 Score : ', (f1_score(y_test, y_pred))) print('Precision Score : ', (precision_score(y_test, y_pred))) print('Recall Score : ', (recall_score(y_test, y_pred))) F-1 Score : 0.6370862700228833 Precision Score : 0.7884615384615384 Recall Score : 0.5375817373250883 [80] !pip install joblib </pre> </div>
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	<div> <pre> In [58]: from sklearn.model_selection import GridSearchCV # Define the parameters grid param_grid = { 'n_estimators': [100, 200, 300], # Number of trees in the forest 'max_depth': [None, 10, 20, 30], # Maximum depth of the tree 'min_samples_split': [2, 5, 10], # Minimum number of samples required to split a node 'min_samples_leaf': [1, 2, 4] # Minimum number of samples required at each leaf node } # Initialize Random Forest Classifier rfc = RandomForestClassifier(random_state=0) # Perform GridSearchCV grid_search = GridSearchCV(rfc, param_grid, cv=5, scoring='accuracy') grid_search.fit(X_train, y_train) # Get the best parameters best_params = grid_search.best_params_ print("Best Parameters:", best_params) # Train using the best parameters best_rfc = RandomForestClassifier(**best_params, random_state=0) best_rfc.fit(X_train, y_train) # Predict and evaluate y_pred = best_rfc.predict(X_test) accuracy = accuracy_score(y_test, y_pred) print("Accuracy Score with Best Parameters:", round(accuracy * 100, 2), "%") Best Parameters: {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200} Accuracy Score with Best Parameters: 68.64 % </pre> </div>