

Project Development Phase Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID35546
Project Name	WEB PHISHING DETECTION
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: Linear Regression</p> <p>MAE – Mean Absolute Error: 0.414450904671261</p> <p>, MSE – Mean Squared Error: 0.2944599042452626</p> <p>, RMSE – Root Mean Squared Error: 0.5426415983365656</p> <p>, R2 score - R-Square Error: 0.7000547130388581</p> <p>Classification Model: Naïve Bayes</p> <p>Confusion Matrix - , Accuray Score- & Classification Report -</p>	<pre> In [100]: from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score from sklearn.metrics import confusion_matrix, classification_report In [101]: # MAE y_test, y_pred = y_test.values, y_pred.values mae = mean_absolute_error(y_test, y_pred) print('MAE: {}'.format(mae)) In [102]: # MSE y_test, y_pred = y_test.values, y_pred.values mse = mean_squared_error(y_test, y_pred) print('MSE: {}'.format(mse)) In [103]: # RMSE y_test, y_pred = y_test.values, y_pred.values rmse = np.sqrt(mse) print('RMSE: {}'.format(rmse)) In [104]: # R2 score y_test, y_pred = y_test.values, y_pred.values r2 = r2_score(y_test, y_pred) print('R2 score: {}'.format(r2)) In [105]: # Confusion Matrix y_test, y_pred = y_test.values, y_pred.values cm = confusion_matrix(y_test, y_pred) print('Confusion Matrix: {}'.format(cm)) In [106]: # Classification Report y_test, y_pred = y_test.values, y_pred.values report = classification_report(y_test, y_pred) print('Classification Report: {}'.format(report)) In [107]: # Accuracy Score y_test, y_pred = y_test.values, y_pred.values accuracy = cm[1][1] / (cm[1][1] + cm[1][0]) print('Accuracy Score: {}'.format(accuracy)) </pre>
2.	Tune the Model	Hyperparameter Tuning - gridsearchCV Validation Method -	<pre> In [108]: from sklearn.metrics import mean_squared_error from sklearn.model_selection import GridSearchCV from sklearn.preprocessing import StandardScaler In [109]: # StandardScaler scaler = StandardScaler() X_train_scaled = scaler.fit_transform(X_train) X_test_scaled = scaler.transform(X_test) In [110]: # GridSearchCV param_grid = {'C': [0.1, 1, 10, 100, 1000], 'gamma': [0.001, 0.01, 0.1, 1, 10]} gscv = GridSearchCV(estimator=svm, param_grid=param_grid, scoring='neg_mean_squared_error', cv=5) gscv.fit(X_train_scaled, y_train_scaled) best_params = gscv.best_params_ print('Best parameters: {}'.format(best_params)) In [111]: # Validation Method y_test, y_pred = y_test.values, y_pred.values mse = mean_squared_error(y_test, y_pred) print('MSE: {}'.format(mse)) </pre>