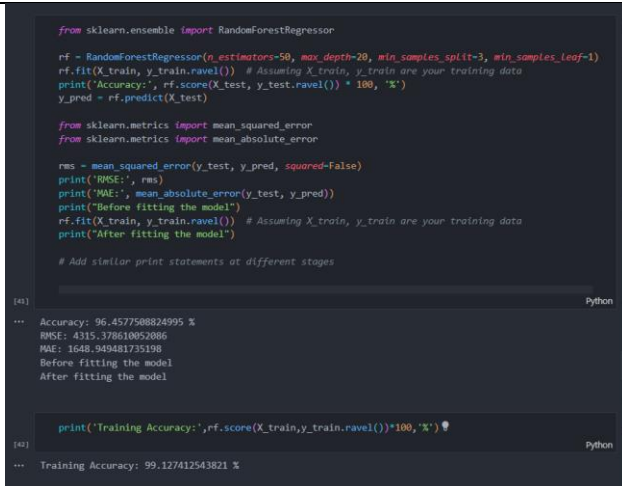
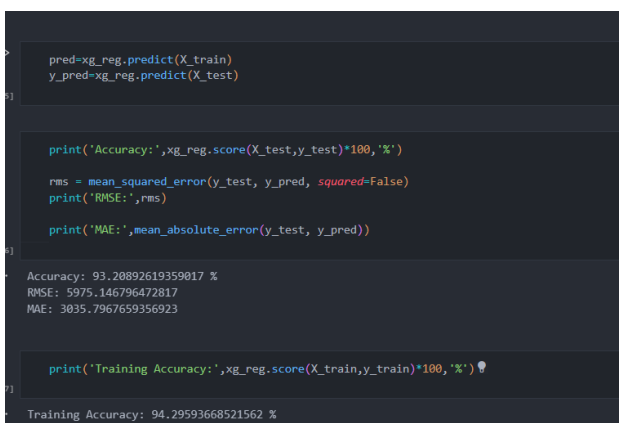


Project Development Phase Model Performance Test

Date	20-11-2023
Team ID	TEAM-591736
Project Name	Walmart store sales forecasting
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> from sklearn.ensemble import RandomForestRegressor rf = RandomForestRegressor(n_estimators=50, max_depth=20, min_samples_split=3, min_samples_leaf=1) rf.fit(X_train, y_train.ravel()) # Assuming X_train, y_train are your training data print("Accuracy:", rf.score(X_test, y_test.ravel()) * 100, "%") y_pred = rf.predict(X_test) from sklearn.metrics import mean_squared_error from sklearn.metrics import mean_absolute_error rms = mean_squared_error(y_test, y_pred, squared=False) print("RMSE:", rms) print("MAE:", mean_absolute_error(y_test, y_pred)) print("Before fitting the model") rf.fit(X_train, y_train.ravel()) # Assuming X_train, y_train are your training data print("After fitting the model") # Add similar print statements at different stages Accuracy: 96.4577588824995 % RMSE: 4315.378610052086 MAE: 1648.949481735198 Before fitting the model After fitting the model print("Training Accuracy:", rf.score(X_train, y_train.ravel()) * 100, "%") Training Accuracy: 99.127412543821 % </pre>  <pre> pred=xg_reg.predict(X_train) y_pred=xg_reg.predict(X_test) print("Accuracy:", xg_reg.score(X_test, y_test) * 100, "%") rms = mean_squared_error(y_test, y_pred, squared=False) print("RMSE:", rms) print("MAE:", mean_absolute_error(y_test, y_pred)) Accuracy: 93.28892619359017 % RMSE: 5975.146796472817 MAE: 3035.7967659356923 print("Training Accuracy:", xg_reg.score(X_train, y_train) * 100, "%") Training Accuracy: 94.29593668521562 % </pre>

2.	Tune the Model	Hyperparameter Tuning - Validation Method -	<pre># Generate forecast forecast = model_auto.arima.predict(n_periods=len(test_data)) forecast = pd.DataFrame(forecast, index=test_data.index, columns=['Predictions']) # Plotting plt.figure(figsize=(20, 6)) plt.title('Prediction of Weekly Sales using Auto ARIMA model', fontsize=20) plt.plot(train_data, Label='Train') plt.plot(test_data, Label='Test') plt.plot(forecast, Label='Prediction using ARIMA Model') plt.legend(loc='best') plt.xlabel('Date', fontsize=14) plt.ylabel('Weekly Sales', fontsize=14) plt.show() # Calculate metrics predictions = forecast['Predictions'] mse = mean_squared_error(test_data, predictions) rmse = np.sqrt(mse) mae = mean_absolute_error(test_data, predictions) print('Mean Squared Error (MSE) of Auto ARIMA:', mse) print('Root Mean Squared Error (RMSE) of Auto ARIMA:', rmse) print('Mean Absolute Deviation (MAE) of Auto ARIMA:', mae)</pre> <div>ARIMA(0,0,0)(0,0,0)[1] intercept : AIC=397.842, Time=0.02 sec ARIMA(0,0,1)(0,0,0)[1] intercept : AIC=399.420, Time=0.09 sec ARIMA(0,0,2)(0,0,0)[1] intercept : AIC=inf, Time=0.18 sec ARIMA(0,0,3)(0,0,0)[1] intercept : AIC=inf, Time=0.20 sec ARIMA(0,0,4)(0,0,0)[1] intercept : AIC=399.645, Time=0.25 sec ARIMA(0,0,5)(0,0,0)[1] intercept : AIC=inf, Time=0.41 sec ARIMA(1,0,0)(0,0,0)[1] intercept : AIC=399.658, Time=0.03 sec ARIMA(1,0,1)(0,0,0)[1] intercept : AIC=401.559, Time=0.06 sec ARIMA(1,0,2)(0,0,0)[1] intercept : AIC=404.248, Time=0.09 sec ARIMA(1,0,3)(0,0,0)[1] intercept : AIC=400.338, Time=0.13 sec ARIMA(1,0,4)(0,0,0)[1] intercept : AIC=inf, Time=0.30 sec ARIMA(2,0,0)(0,0,0)[1] intercept : AIC=399.687, Time=0.10 sec ARIMA(2,0,1)(0,0,0)[1] intercept : AIC=403.287, Time=0.20 sec ARIMA(2,0,2)(0,0,0)[1] intercept : AIC=404.649, Time=0.33 sec ARIMA(2,0,3)(0,0,0)[1] intercept : AIC=402.435, Time=0.41 sec ARIMA(2,0,4)(0,0,0)[1] intercept : AIC=inf, Time=0.41 sec ARIMA(3,0,0)(0,0,0)[1] intercept : AIC=400.869, Time=0.29 sec ARIMA(2,0,0)(0,0,0)[1] intercept : AIC=399.687, Time=0.10 sec ARIMA(2,0,1)(0,0,0)[1] intercept : AIC=403.287, Time=0.20 sec ARIMA(2,0,2)(0,0,0)[1] intercept : AIC=404.649, Time=0.33 sec ARIMA(2,0,3)(0,0,0)[1] intercept : AIC=402.435, Time=0.41 sec ARIMA(3,0,0)(0,0,0)[1] intercept : AIC=400.869, Time=0.29 sec ARIMA(3,0,1)(0,0,0)[1] intercept : AIC=403.022, Time=0.22 sec ARIMA(3,0,2)(0,0,0)[1] intercept : AIC=405.264, Time=0.33 sec ARIMA(4,0,0)(0,0,0)[1] intercept : AIC=402.820, Time=0.17 sec ARIMA(4,0,1)(0,0,0)[1] intercept : AIC=405.025, Time=0.41 sec ARIMA(5,0,0)(0,0,0)[1] intercept : AIC=404.311, Time=0.15 sec</div> <div>Best model: ARIMA(0,0,0)(0,0,0)[1] intercept Total Fit time: 4.405 seconds</div> <div></div> <div>Mean Squared Error (MSE) of Auto ARIMA: 468477.9539606969 Root Mean Squared Error (RMSE) of Auto ARIMA: 684.4544937106461 Mean Absolute Deviation (MAE) of Auto ARIMA: 446.8196095294599</div>
----	----------------	---	---