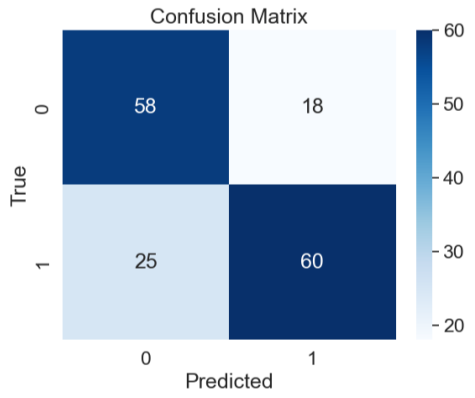
Project Development Phase
Model Performance Test

Date	10 November 2022
Team ID	Team - 519680
Project Name	Diabetes 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	**Regression Model:** MSE – 0.26708, RMSE – 0.516798, **Classification Model:** Confusion Matrix Accuray Score-0.695 & Classification Report	**Random Forest** `# Check MSE & RMSE` `mse =mean_squared_error(Y_test, y_pred)` `print('Mean Squared Error : '+ str(mse))` `rmse = math.sqrt(mean_squared_error(Y_test, y_pred))` `print('Root Mean Squared Error : '+ str(rmse))` Mean Squared Error : 0.2670807453416149 Root Mean Squared Error : 0.5167985539275578 `matrix = classification_report(Y_test,y_pred )` `print(matrix)` precision recall f1-score support 0 0.70 0.76 0.73 76 1 0.77 0.71 0.74 85 accuracy 0.73 161 macro avg 0.73 0.73 0.73 161 weighted avg 0.74 0.73 0.73 161

			`` def plot_confusion_matrix(y_true, y_pred): cm = confusion_matrix(y_true, y_pred) sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=True) plt.xlabel('Predicted') plt.ylabel('True') plt.title('Confusion Matrix') plt.show() # Assuming Y_test and y_pred are defined plot_confusion_matrix(Y_test, y_pred) ``` 
2.	Tune the Model	Hyperparameter Tuning - GridSearchCV Validation Method - Train-Test Split	``` X_train , X_test , Y_train , Y_test = train_test_split(x_sm,y_sm, test_size=0.3 , random_state=42) ``` ``` for x in [X_train, X_test, Y_train, Y_test]: print(len(x)) ``` ``` 375 161 375 161 ``` ``` for i,j in grid_models: grid = GridSearchCV(estimator=i,param_grid = j, scoring = 'accuracy',cv=2) grid.fit(X_train, Y_train) best_accuracy = grid.best_score_ best_param = grid.best_params_ print('{}:\nBest Accuracy : {:.2f}%'.format(i,best_accuracy*100)) print('Best Parameters : ',best_param) print('') print('----------------') print('') ``` ``` KNeighborsClassifier(): Best Accuracy : 70.93% Best Parameters : {'algorithm': 'auto', 'n_neighbors': 10, 'weights': 'distance'} ---------------- DecisionTreeClassifier(): Best Accuracy : 70.66% Best Parameters : {'criterion': 'entropy', 'max_depth': 8, 'min_samples_leaf': 4} ---------------- RandomForestClassifier(): Best Accuracy : 74.66% Best Parameters : {'criterion': 'entropy', 'max_depth': 13, 'max_features': 3, 'n_estimators': 50} ---------------- ``` ``` # Compare accuracies for i,model in enumerate(pipelines): print("{} Test Accuracy: {}".format(pipe_dict[i],model.score(X_test,Y_test))) ``` ``` Logistic Regression Test Accuracy: 0.6832298136645962 Decision Tree Test Accuracy: 0.6086956521739131 Random Forest Test Accuracy: 0.6956521739130435 ```