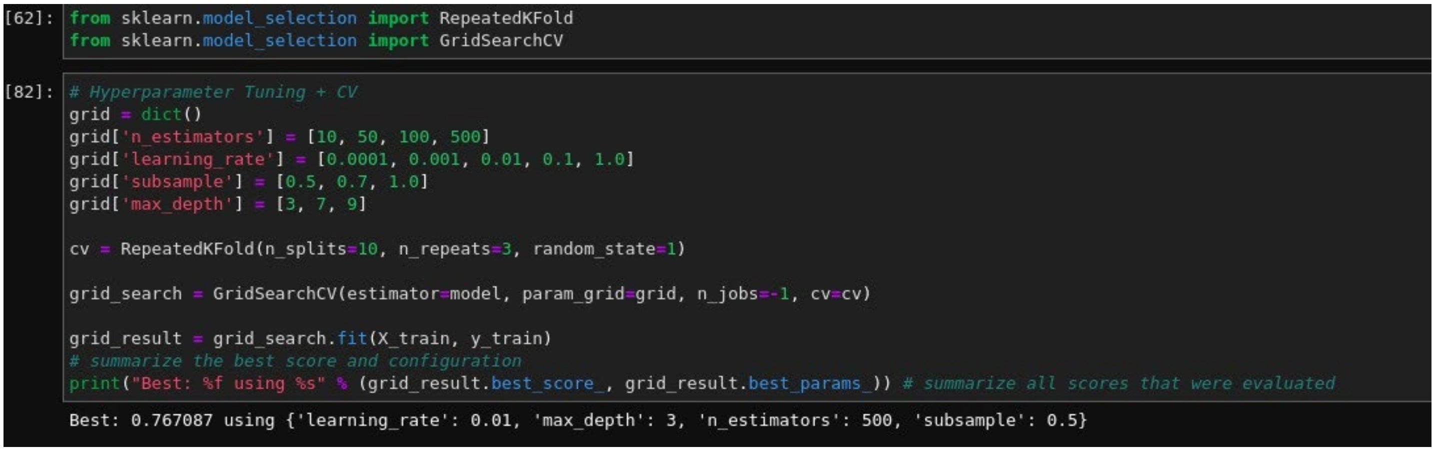
Project Development Phase
Model Performance Test

Date	14 November 2022
Team ID	PNT2022TMID16260
Project Name	Project – University Admit Eligibility Predictor
Maximum Marks	10 Marks

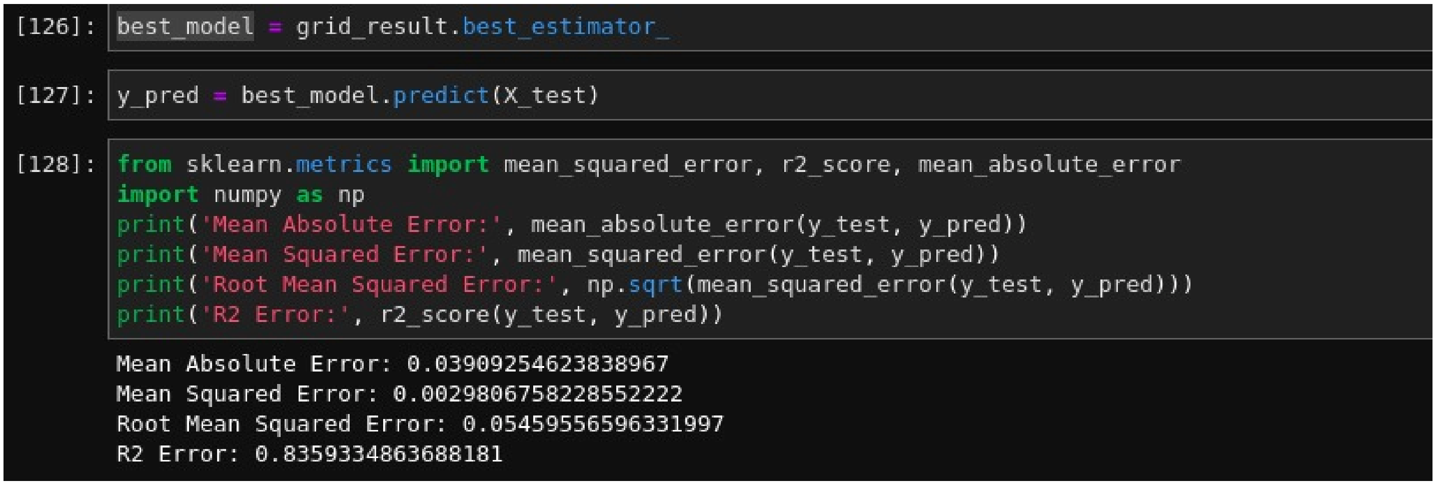
Model Performance Testing:

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

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE -, MSE -, RMSE -, R2 score - **Classification Model:** Confusion Matrix -, Accuracy Score- & Classification Report -	**Regression Model:** **Mean Absolute Error (MAE) –** 0.390254623838967 **Mean Squared Error (MSE) –** 0.0029806758228552222 **Root Mean Squared Error (RMSE) –** 0.05459556596331997 **R2 Score –** 0.835933486388181
2.	Tune the Model	**Hyperparameter Tuning –** GridSearchCv with Repeated 10Folds is used to find the set of hyperparameters for the given training set. **Validation Method -**	



```
[62]: from sklearn.model_selection import RepeatedKFold
      from sklearn.model_selection import GridSearchCV

[82]: # Hyperparameter Tuning + CV
      grid = dict()
      grid['n_estimators'] = [10, 50, 100, 500]
      grid['learning_rate'] = [0.0001, 0.001, 0.01, 0.1, 1.0]
      grid['subsample'] = [0.5, 0.7, 1.0]
      grid['max_depth'] = [3, 7, 9]

      cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

      grid_search = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1, cv=cv)

      grid_result = grid_search.fit(X_train, y_train)
      # summarize the best score and configuration
      print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_)) # summarize all scores that were evaluated

      Best: 0.767087 using {'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 500, 'subsample': 0.5}
```



```
[126]: best_model = grid_result.best_estimator_

[127]: y_pred = best_model.predict(X_test)

[128]: from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
       import numpy as np
       print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
       print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
       print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
       print('R2 Error:', r2_score(y_test, y_pred))

       Mean Absolute Error: 0.03909254623838967
       Mean Squared Error: 0.0029806758228552222
       Root Mean Squared Error: 0.05459556596331997
       R2 Error: 0.8359334863688181
```