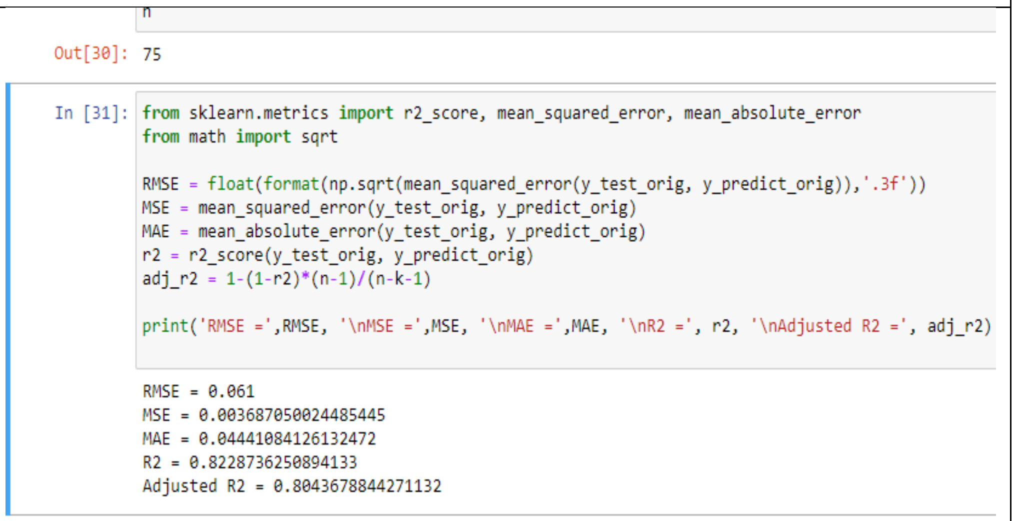
Project Development Phase
Model Performance Test

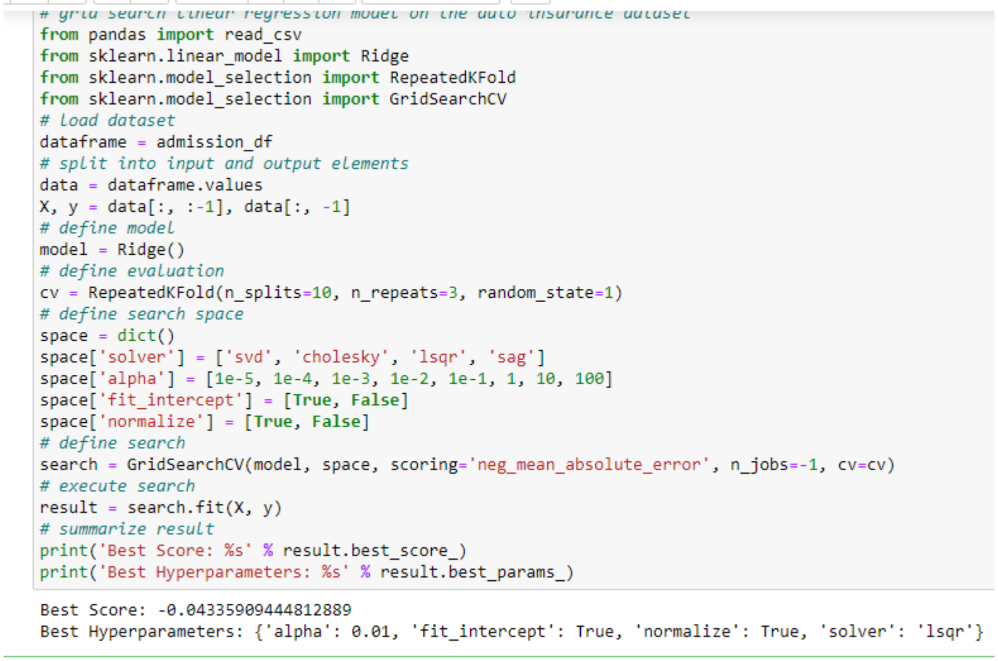
Date	10 November 2022
Team ID	PNT2022TMID18312
Project Name	Project – University Admit Eligibility Predictor
Maximum Marks	10 Marks

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 –0.04441084126132472, MSE –0.003687050024485445, RMSE - 0.061, R2 score -0.8228736250894133	

```
n

Out[30]: 75

In [31]: from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
         from math import sqrt

         RMSE = float(format(np.sqrt(mean_squared_error(y_test_orig, y_predict_orig)),'.3f'))
         MSE = mean_squared_error(y_test_orig, y_predict_orig)
         MAE = mean_absolute_error(y_test_orig, y_predict_orig)
         r2 = r2_score(y_test_orig, y_predict_orig)
         adj_r2 = 1-(1-r2)*(n-1)/(n-k-1)

         print('RMSE =',RMSE, '\nMSE =',MSE, '\nMAE =',MAE, '\nR2 =', r2, '\nAdjusted R2 =', adj_r2)

         RMSE = 0.061
         MSE = 0.003687050024485445
         MAE = 0.04441084126132472
         R2 = 0.8228736250894133
         Adjusted R2 = 0.8043678844271132
```

2.	Tune the Model	Hyperparameter Tuning - {'alpha': 0.01, 'fit_intercept': True, 'normalize': True, 'solver': 'lsqr'} Best Score - 0.04335909444812889	

```
# grid search linear regression model on the auto insurance dataset
from pandas import read_csv
from sklearn.linear_model import Ridge
from sklearn.model_selection import RepeatedKFold
from sklearn.model_selection import GridSearchCV
# load dataset
dataframe = admission_df
# split into input and output elements
data = dataframe.values
X, y = data[:, :-1], data[:, -1]
# define model
model = Ridge()
# define evaluation
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
# define search space
space = dict()
space['solver'] = ['svd', 'cholesky', 'lsqr', 'sag']
space['alpha'] = [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100]
space['fit_intercept'] = [True, False]
space['normalize'] = [True, False]
# define search
search = GridSearchCV(model, space, scoring='neg_mean_absolute_error', n_jobs=-1, cv=cv)
# execute search
result = search.fit(X, y)
# summarize result
print('Best Score: %s' % result.best_score_)
print('Best Hyperparameters: %s' % result.best_params_)
```

```
Best Score: -0.04335909444812889
Best Hyperparameters: {'alpha': 0.01, 'fit_intercept': True, 'normalize': True, 'solver': 'lsqr'}
```