

## Model Optimization and Tuning Phase

Date	07 July 2024
Team ID	739899
Project Title	SmartLender – Envisioning Success: Predicting University Scores With Machine Learning
Maximum Marks	2Marks

### Performance Metrics Comparison Report (2 Marks):

A Performance Metrics Comparison Report systematically evaluates the effectiveness of various machine learning models or algorithms by comparing key metrics such as MAE, MSE, R-Square. This report highlights the strengths and weaknesses of each model, providing insights into their performance on different datasets or tasks. By presenting a clear, detailed analysis, the report aids in selecting the most suitable model for deployment, ensuring optimal performance in realworld applications.



```
[56] # Assuming 'x_test' is available in the environment and is a pandas DataFrame or a NumPy array.
      y_pred = linReg.predict(x_test) # Predict on the entire x_test dataset

      print("Prediction Evaluation using Linear Regression")
      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('R-squared:', r2_score(y_test, y_pred))
```

Prediction Evaluation using Linear Regression  
 Mean Absolute Error: 0.9264657671450711  
 Mean Squared Error: 1.7890643253785259  
 Root Mean Squared Error: 1.337559092294066  
 R-squared: 0.7439493774592185

```
y_pred = lassoReg.predict(x_test)
print("Prediction Evaluation using lasso Regression")
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('R-squared:', r2_score(y_test, y_pred))
```

Prediction Evaluation using lasso Regression  
 Mean Absolute Error: 0.9352851280381133  
 Mean Squared Error: 1.7857764808364731  
 Root Mean Squared Error: 1.3363294806433303  
 R-squared: 0.7444199332854502

```
y_pred = svr.predict(x_test)
print("Prediction Evaluation using support vector Regression")
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('R-squared:', r2_score(y_test, y_pred))
```

Prediction Evaluation using support vector Regression  
 Mean Absolute Error: 0.5454340693726399  
 Mean Squared Error: 1.2411917299771091  
 Root Mean Squared Error: 1.1140878466158355  
 R-squared: 0.8223608225568596

[+ Code](#)
[+ Text](#)

```
[59] y_pred = dt.predict(x_test)
print("Prediction Evaluation using Decision Regression ")
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('R-squared:', r2_score(y_test, y_pred))
```

Prediction Evaluation using Decision Regression  
 Mean Absolute Error: 5.264475724040743e-15  
 Mean Squared Error: 2.7561365735867205e-28  
 Root Mean Squared Error: 1.6601616106833456e-14  
 R-squared: 1.0

```
✓ 0s ▶ y_pred = rf.predict(x_test)
print("Prediction Evaluation using Random Regression")
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('R-squared:', r2_score(y_test, y_pred))
```

```
⇌ Prediction Evaluation using Random Regression
Mean Absolute Error: 0.010686590909090909
Mean Squared Error: 0.0009053592244319952
Root Mean Squared Error: 0.0300891878327082
R-squared: 0.9998704251212489
```