



# **Model Development Phase**

Date	06 July 2024
Team ID	739899
Project Title	SmartLender – Envisioning Success: Predicting University Scores With Machine Learning
Maximum Marks	4 Marks

### Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

## **Initial Model Training Code:**

[38] y\_pred = linReg.predict(x\_test)

```
linReg = LinearRegression()
linReg.fit(x_train,y_train)

LinearRegression
LinearRegression()
```





```
/<sub>1s</sub> [40]
         lassoReg = linear_model.Lasso(alpha = 0.1)
         lassoReg.fit(x,y)
   ₹
                Lasso
         Lasso(alpha=0.1)
  [41] y_pred = lassoReg.predict(x_test)
\bigvee_{0s} [43] svr = SVR().fit(x,y)
os [44]
         y_pred = svr.predict(x_test)
  [46] dt = DecisionTreeRegressor(random_state = 0)
        dt.fit(x,y)
   ₹
                 DecisionTreeRegressor
        DecisionTreeRegressor(random state=0)
```





```
f = RandomForestRegressor(n_estimators = 100 , random_state = 0)
    rf.fit(x,y)

RandomForestRegressor
RandomForestRegressor(random_state=0)

y_pred = rf.predict(x_test)
```

### **Model Validation and Evaluation Report:**

Model	Summary	Training and Validation Performance Metrics
Linear Regression	A commany is rect in manifolds but the environment and it a name introduce on a bandy error present lawary frequency. The dutation for department of the environment of the environme	Prediction Evaluation using Linear Regression Plean Absolute Error: 0.926467971459713 Plean Squared Error: 1.7896643253785259 Boot Plean Squared Error: 1.11255880229M066. R. squared: 0.7459493774592185
Lasso Regression	[37]  y_prod = istoineq.gendict(x_test)  mint(*residetine tomination using land tegression*)  mint(*men desidet cornt*, every shouldes error(y_test, y_pred))  mint(*men *Squared irror**, every shouldes error(y_test, y_pred))  mint(*Men *Men *Langrad irror**, every should error(y_test, y_pred))  mint(*Men *Men *Langrad irror**, ex_short(sex_supered error(y_test, y_pred)))  mint(*Neutror**, ry_urare(y_test, y_pred))	Prediction Evaluation using lasso Regression Mean Absolute Error: 0.9959851280031133 Nexan Squared Frore: 1.052764868334731 Root Mean Squared Error: 1.3363294886433383 R-squared: 0.7444299332804582
Support Vector Machine	<pre> (m) y_prod = ser_gradict(x_bist) print(Twoduction foundation using unggest vector Regression*) print(Twoduction(S_traver), mon_disolone_print(y_text, y_prod)) print(Twoduction(y_text), mon_disolone_print(y_text, y_prod)) print(Twoduction(y_text), new_disolone_print(y_text, y_prod)) print(Twoduction(y_text, y_prod)) print(Twoduction(y_text, y_prod)) </pre>	Prediction Evaluation using support sector Regression  Pean Abjoints Error: 0.5454304690726590  Pean Squared Error: E.Malient/Pen/72691  Boot Pean-Squared Error: 2.130007050055  8-squared: 0.822800070506590





# Decision Tree [59] y pred = dt.predict(y, test) wrist("New Recolute Error:", mean squared error(y test, y pred)) wrist("New Accolute Error:", mean squared error(y test, y pred)) wrist("New Accolute Error:", mean squared error(y test, y pred)) wrist("New Accolute Error: ", mean squared error(y test, y pred)) wrist("New Accolute Error: 2, 75617657754667430-15 Hean Squared: Error: 2, 7561765775467450-14 Resquared: 1.8

Random		
Forest	[60] y_pred = rf.gredict(x_test)  print("Frediction Evaluation using Random Repression")  print("Mean Absolute Error:", mean absolute error(y_test, y_pred))  print("Mean Squared Error:", mean squared error(y_test, y_pred))  print("Mean Squared Error:", rp.sqrt(mean squared error(y_test, y_pred)))  print("Mean squared:", r2_score(y_test, y_pred))	Prediction Evaluation using Random Regression Mean Absolute Error: 0.010605909090909049 Mean Squared Error: 0.0090953592244319952 Moot Mean Squared Error: 0.0300091878127082 R-squared: 0.9998764251212489