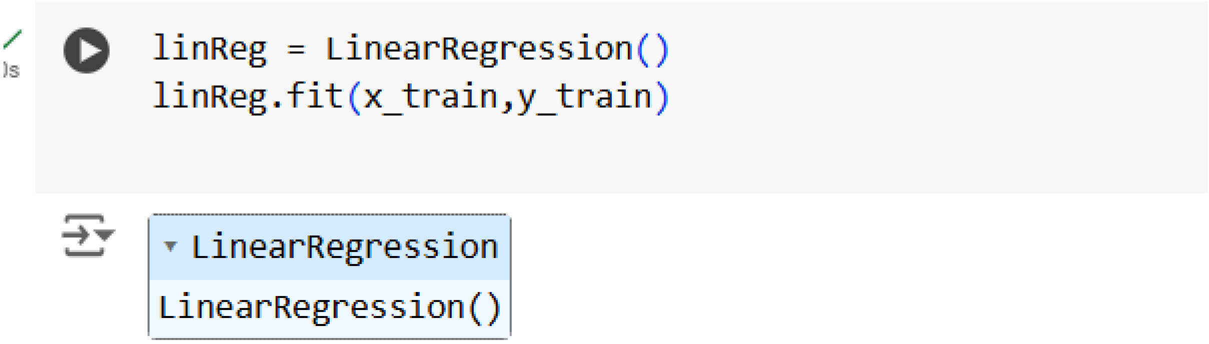
Model Development Phase

Date	06 July 2024
Team ID	73973
Project Title	SmartLender – Envisioning Success: Predicting University Scores With Machine Learning
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

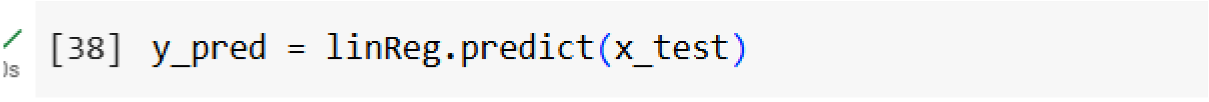
The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Initial Model Training Code :

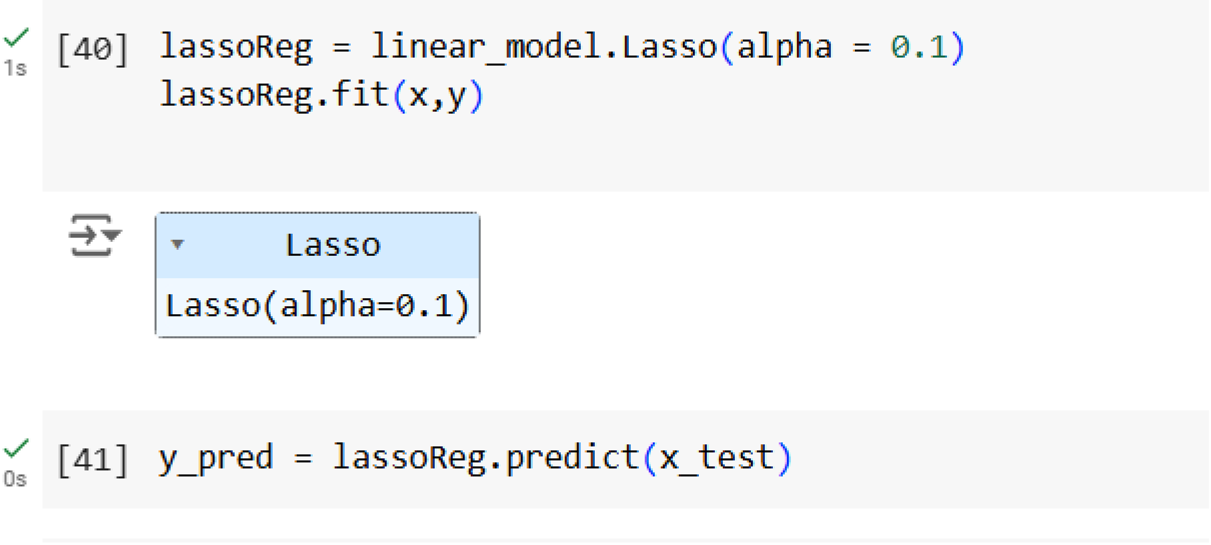


```
linReg = LinearRegression()
linReg.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```



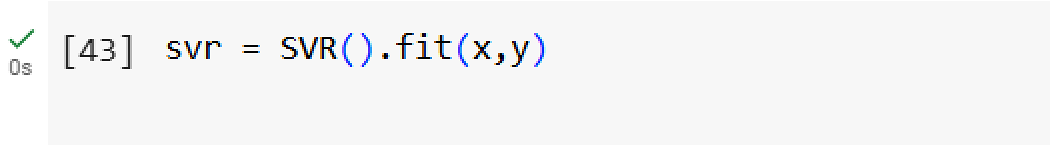
```
[38] y_pred = linReg.predict(x_test)
```



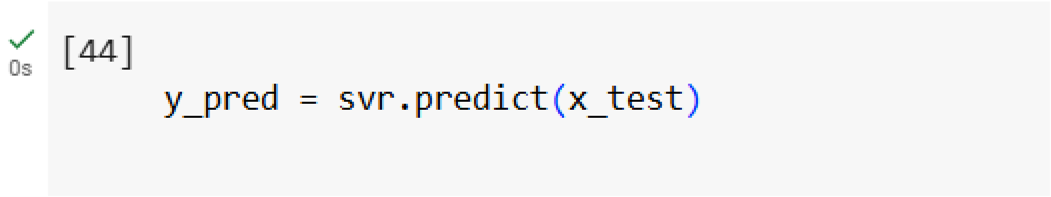
```
[40] lassoReg = linear_model.Lasso(alpha = 0.1)
     lassoReg.fit(x,y)
```

```
Lasso
Lasso(alpha=0.1)
```

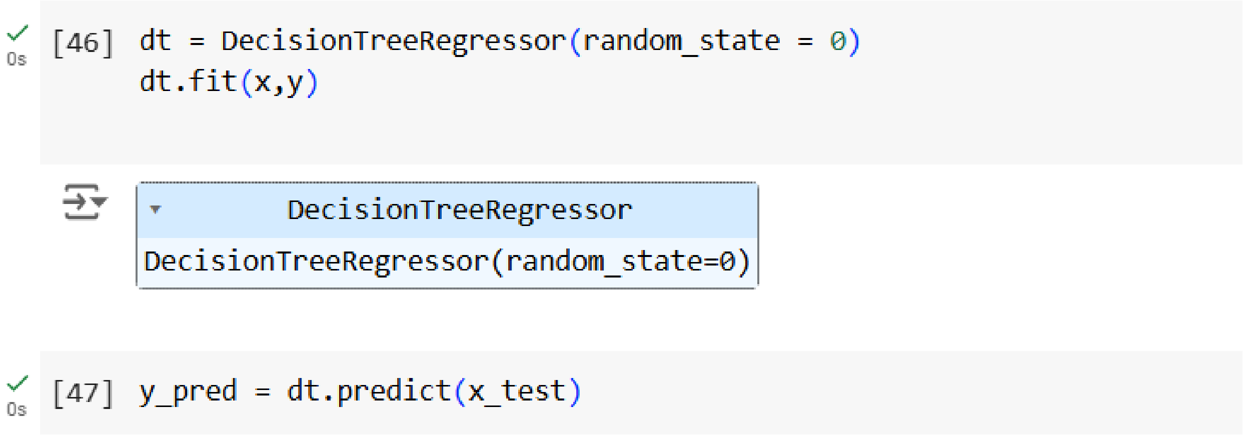
```
[41] y_pred = lassoReg.predict(x_test)
```



```
[43] svr = SVR().fit(x,y)
```



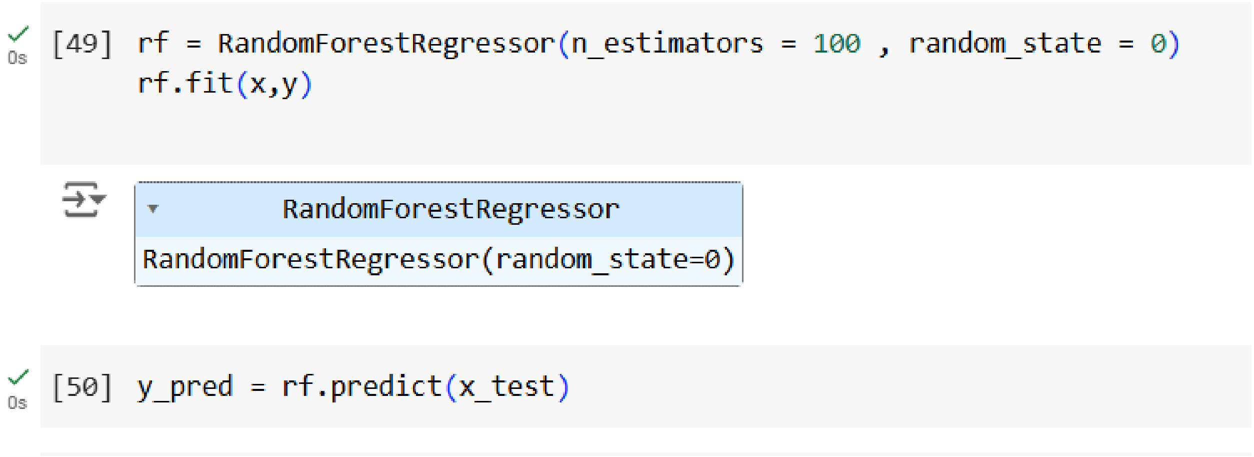
```
[44]
      y_pred = svr.predict(x_test)
```



```
[46] dt = DecisionTreeRegressor(random_state = 0)
     dt.fit(x,y)
```

```
DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

```
[47] y_pred = dt.predict(x_test)
```



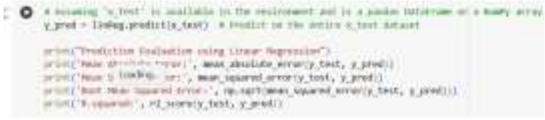
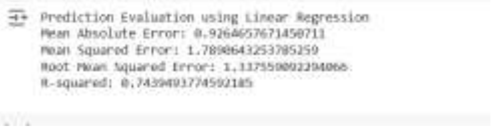
```
[49] rf = RandomForestRegressor(n_estimators = 100 , random_state = 0)
     rf.fit(x,y)
```

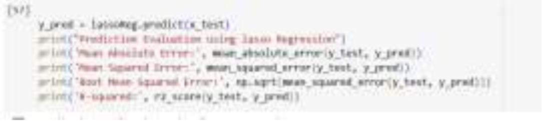
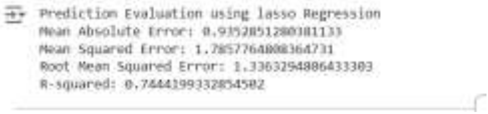
```
RandomForestRegressor
RandomForestRegressor(random_state=0)
```

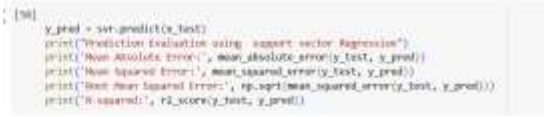
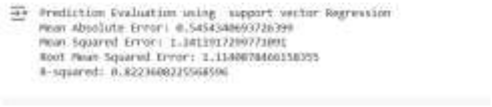
```
[50] y_pred = rf.predict(x_test)
```

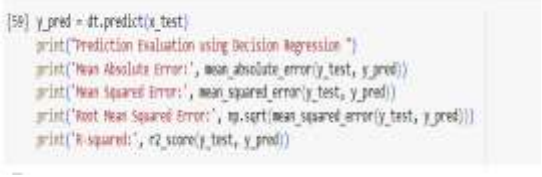
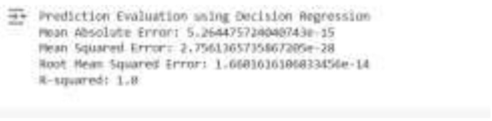
Model Validation and Evaluation Report :

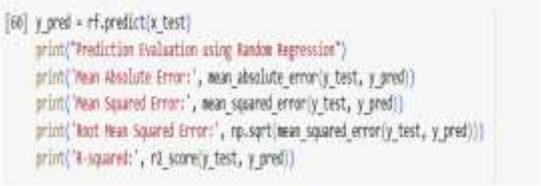
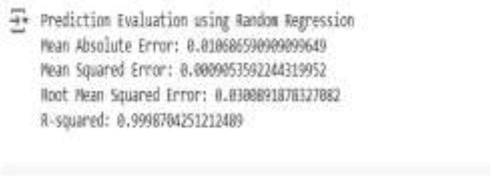
Model	Summary	Training and Validation Performance Metrics

Linear Regression	
	# 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

Lasso Regression	
	[57]
	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: 1.3363294886433303
		R-squared: 0.7444199332854502

Support Vector Machine	
	[58]
	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: 1.2411917209773891
		Root Mean Squared Error: 1.1140078460158355
		R-squared: 0.8223608225568596

Decision Tree	
	[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

Random Forest	
	`[60] 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: 0.010686590909099649`
		`Mean Squared Error: 0.0009053592244319952`
		`Root Mean Squared Error: 0.0300891878327082`
		`R-squared: 0.9998704251212489`