Sri Sivasubramaniya Nadar College of Engineering, Chennai
(An autonomous Institution affiliated to Anna University)

Degree & Branch	M. Tech (Integrated) Computer Science & Engineering	Semester	V
Subject Code & Name	ICS1512 & Machine Learning Algorithms Laboratory		
Academic year	2025-2026 (Odd)	Batch:2023-2028	**Due date:**

Experiment 2: Loan Amount Prediction using Linear Regression

Objective

Apply Linear Regression to predict the loan amount sanctioned to users using the dataset provided. Visualize and interpret the results to gain insights into the model performance.

Dataset

Download the dataset from the Kaggle repository: Predict Loan Amount Data – Kaggle. The dataset contains historical records of loan amounts sanctioned to users, along with various features. The goal is to use these features to predict the sanctioned loan amount.

Task Description

Develop a Python program using the `Scikit-learn` library to build and evaluate a Linear Regression (LR) model for loan amount prediction. Use `Matplotlib` to visualize key insights and results.

Implementation Steps

1. **Load the dataset.**
2. **Pre-process the data:**
   - Handle missing values
   - Encode categorical variables
   - Normalize or standardize the features
3. **Perform Exploratory Data Analysis (EDA)** to understand the distributions and relationships in the dataset.
4. **Apply feature engineering techniques** to improve model performance.
5. **Split the dataset** into training, testing, and validation sets.
6. **Train the Linear Regression model** on the training set.
7. **Evaluate the model** on the testing and validation sets.

8. **Measure performance** using metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R^2 score.

9. **Visualize the results:**
   - Plot predicted vs actual values
   - Visualize feature importance or coefficients

Important Plots to Include

- **Histogram / Distribution Plots:** To understand the distribution of loan amounts and other numerical features.
- **Scatter Plots:** To examine the relationship between key features (e.g., income, credit score) and the loan amount.
- **Correlation Heatmap:** To identify multicollinearity and relationships among features.
- **Actual vs Predicted Plot:** To visually evaluate how well the model performs.
- **Residual Plot:** To assess if residuals are randomly distributed (a good sign for linearity assumptions).
- **Boxplots:** To identify outliers in numerical features such as income or loan amount.
- **Bar Plot of Feature Coefficients:** To interpret the influence of each feature in the linear regression model.

Results Summary Table

Students are expected to fill in the following table based on their model's performance and visualizations.

Cross-Validation Results Table

If you have used K-Fold Cross-Validation (e.g., with $K = 5$), report the evaluation metrics for each fold in the table below.

Table 1: Cross-Validation Results (K = _)

Fold	**MAE**	**MSE**	**RMSE**	**R^2 Score**
Fold 1				
Fold 2				
Fold 3				
Fold 4				
Fold 5				
Average				

Results Summary Table

Students are expected to fill in the following table based on their model's performance and visualizations.

Results Summary Table

Students are expected to fill in the following table based on their model's performance and visualizations.

Table 2: Summary of Results for Loan Amount Prediction

Description	**Student's Result**
Dataset Size (after preprocessing)	
Train/Test Split Ratio	
Feature(s) Used for Prediction	
Model Used	Linear Regression
Cross-Validation Used? (Yes/No)	
If Yes, Number of Folds (K)	
Reference to CV Results Table	Table 1
Mean Absolute Error (MAE) on Test Set	
Mean Squared Error (MSE) on Test Set	
Root Mean Squared Error (RMSE) on Test Set	
R^2 Score on Test Set	
Adjusted R^2 Score on Test Set	
Most Influential Feature(s)	
Observations from Residual Plot	
Interpretation of Predicted vs Actual Plot	
Any Overfitting or Underfitting Observed?	
If Yes, Brief Justification (e.g., training vs test error, residual patterns)	

Observation Notes

1. **Aim**
2. **Libraries Used**
3. **Objective**

4. **Tables Included** *(e.g., Results Summary Table, Cross-Validation Results Table)*

5. **Learning Outcomes and Best Practices**

Report

1. **Aim**

2. **Libraries Used**

3. **Objective**

4. **Mathematical Description**

5. **Code with Plot**

```
# Code goes here inside verbatim environment
```

6. **Included Plots**

7. **Results Tables**

8. **Best Practices**

9. **Learning Outcomes**

References

- Scikit-learn: LinearRegression API

- INRIA: Linear Regression in Scikit-learn

- StackAbuse: Linear Regression with Scikit-learn

Optional Task

Implement Linear Regression Without Using Scikit-learn

As an optional extension, implement the Linear Regression model **from scratch**, without using the `Scikit-learn` library. This task will help you understand the internal workings of the algorithm, including the derivation of coefficients using:

- **Normal Equation:**
$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- **Gradient Descent:** Update weights iteratively to minimize the Mean Squared Error.

Tasks:

1. Implement Linear Regression using NumPy (or manual matrix operations).

2. Train and test your model on the same dataset.

3. Compare the performance metrics (MAE, MSE, RMSE, R^2) with the Scikit-learn implementation.

4. Briefly comment on the differences (if any).

Note: You may still use Matplotlib, Pandas, and NumPy for data handling and visualization. Avoid using high-level ML libraries like Scikit-learn, TensorFlow, or PyTorch for model implementation.