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

Degree & Branch	M.Tech - 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

Aim :
To Develop and evaluate the Linear Regression model for predicting the loan amount sanctioned using a set of independent features and visualize the results.

Libraries used :
- Numpy
- Pandas
- Scikit-learn
- Matplotlib
- Seaborn

Mathematical / Theoretical Description :
Linear Regression is a supervised learning algorithm used for predicting a continuous dependent variable y based on one or more independent variables (features) $x_1, x_2, \ldots, x_n$.

1. Hypothesis Function:
For n input features, the hypothesis function is:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Where:

- $\hat{y}$ is the predicted output (loan amount)
- β_0 is the intercept (bias term)
- $\beta_1, \ldots, \beta_n$ are the coefficients (weights) of the features

2. Cost Function:
To measure the error between predicted values $\hat{y}_i$ and actual values y_i, we use the Mean Squared Error (MSE):

$$J(\beta) = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

3. **Evaluation Metrics:**

- **Mean Absolute Error (MAE)**:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

- **Root Mean Squared Error (RMSE)**:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

- **R-squared Score** (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  Measures the proportion of variance in the dependent variable explained by the model.

- **Adjusted** R^2:

$$R^2_{adj} = 1 - (1 - R^2)\cdot\frac{n-1}{n-p-1}$$

  Adjusts R^2 for the number of predictors p and observations n.

Implementation Steps :

1. Load and preprocess the dataset (handle missing values, encode categorical variables, scale features).
2. Perform Exploratory Data Analysis (EDA).
3. Split the dataset into training, testing, and validation sets.
4. Train the Linear Regression model.
5. Evaluate the model using relevant metrics.
6. Perform K-Fold Cross-Validation.
7. Visualize results: predicted vs actual, residuals, and feature coefficients.

Code :

```
from google.colab import drive
drive.mount('/content/drive')

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.impute import SimpleImputer
```

```python
data = pd.read_csv('/content/drive/My Drive/Sem5/ml/a2/train.csv')
data.drop(['Name','Customer ID','Gender','Age','Property ID',
           'Expense Type 1','Expense Type 2','Type of Employment',
           'Has Active Credit Card','Co-Applicant'], axis=1, inplace=True)
```

```python
# Impute Profession
sns.histplot(data['Profession'], kde=True)
plt.xticks(rotation=90)
plt.show()

# Correlation matrix
df_numeric = data.select_dtypes(include='number')
corr_matrix = df_numeric.corr()
sns.heatmap(corr_matrix)

# Property Age Imputation
prop_age_imputer = SimpleImputer(strategy='mean')
data['Property Age'] = prop_age_imputer.fit_transform(data[['Property Age'
    ]])

# Income Imputation
income_imputer = SimpleImputer(strategy='mean')
data['Income (USD)'] = income_imputer.fit_transform(data[['Income (USD)'
    ]])

# Income Stability
income_stab_imputer = SimpleImputer(strategy='most_frequent')
data[['Income Stability']] = income_stab_imputer.fit_transform(data[['
    Income Stability']])

# Current Loan Expenses
cur_loan_exp_imputer = SimpleImputer(strategy='mean')
data['Current Loan Expenses (USD)'] = cur_loan_exp_imputer.fit_transform(
    data[['Current Loan Expenses (USD)']])

# Credit Score
credit_score_imputer = SimpleImputer(strategy='mean')
data['Credit Score'] = credit_score_imputer.fit_transform(data[['Credit
    Score']])

# Dependents
dependents_imputer = SimpleImputer(strategy='median')
data['Dependents'] = dependents_imputer.fit_transform(data[['Dependents'
    ]])

# Property Location
prop_loc_imputer = SimpleImputer(strategy='most_frequent')
data[['Property Location']] = prop_loc_imputer.fit_transform(data[['
    Property Location']])

# Drop rows with missing target
data.dropna(subset=['Loan Sanction Amount (USD)'], inplace=True)
```

```python
# Encoding categorical variables
data = pd.get_dummies(data, columns=['Income Stability'], drop_first=True)
data = pd.get_dummies(data, columns=['Location', 'Property Location'],
    drop_first=True)

# Profession Encoding
target_mean = data.groupby('Profession')['Loan Sanction Amount (USD)'].
    mean()
data['Profession_encoded'] = data['Profession'].map(target_mean)
data.drop(columns=['Profession'], inplace=True)

# Outlier handling
outlier_column = ['Income (USD)', 'Loan Amount Request (USD)',
       'Current Loan Expenses (USD)', 'Dependents', 'Credit Score',
       'Property Age', 'Property Type',
       'Property Price','Profession_encoded']

for col in outlier_column:
  Q1 = data[col].quantile(0.25)
  Q3 = data[col].quantile(0.75)
  IQR = Q3 - Q1
  low = Q1 - 1.5 * IQR
  high = Q3 + 1.5 * IQR
  data[col] = data[col].clip(low, high)

# Scaling
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
numeric_cols = ['Income (USD)', 'Loan Amount Request (USD)',
       'Current Loan Expenses (USD)', 'Dependents', 'Credit Score',
       'Property Age', 'Property Type',
       'Property Price','Profession_encoded']
data[numeric_cols] = scaler.fit_transform(data[numeric_cols])

# Prepare X and y
target = data[['Loan Sanction Amount (USD)']]
data.drop(columns=['Loan Sanction Amount (USD)'], inplace=True)
x = data
y = target
```

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
    r2_score

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,
    random_state=42)
lr_model = LinearRegression()
lr_model.fit(x_train,y_train)
y_pre_train = lr_model.predict(x_train)

# Training Metrics
print("Training Accuracy :",lr_model.score(x_train,y_train)*100)
print("MAE :", mean_absolute_error(y_train,y_pre_train))
```

```
print("MSE :", mean_squared_error(y_train,y_pre_train))
print("R2 :", r2_score(y_train,y_pre_train))

# Testing Metrics
y_pre_test = lr_model.predict(x_test)
print("Testing Accuracy :",lr_model.score(x_test,y_test)*100)
print("MAE :", mean_absolute_error(y_test,y_pre_test))
print("MSE :", mean_squared_error(y_test,y_pre_test))
print("R2 :", r2_score(y_test,y_pre_test))

# KFold Cross Validation
from sklearn.model_selection import KFold
kf = KFold(n_splits=5, shuffle=True, random_state=42)
model = LinearRegression()

mse_scores = []
mae_scores = []
r2_scores = []

for train_index, test_index in kf.split(x):
    x_train, x_test = x.iloc[train_index], x.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)

    mse_scores.append(mean_squared_error(y_test, y_pred))
    mae_scores.append(mean_absolute_error(y_test, y_pred))
    r2_scores.append(r2_score(y_test, y_pred))

print("Cross Validation Results:")
print("Mean MAE:", np.mean(mae_scores))
print("Mean MSE:", np.mean(mse_scores))
print("Mean R2:", np.mean(r2_scores))
```

Residual Plots :

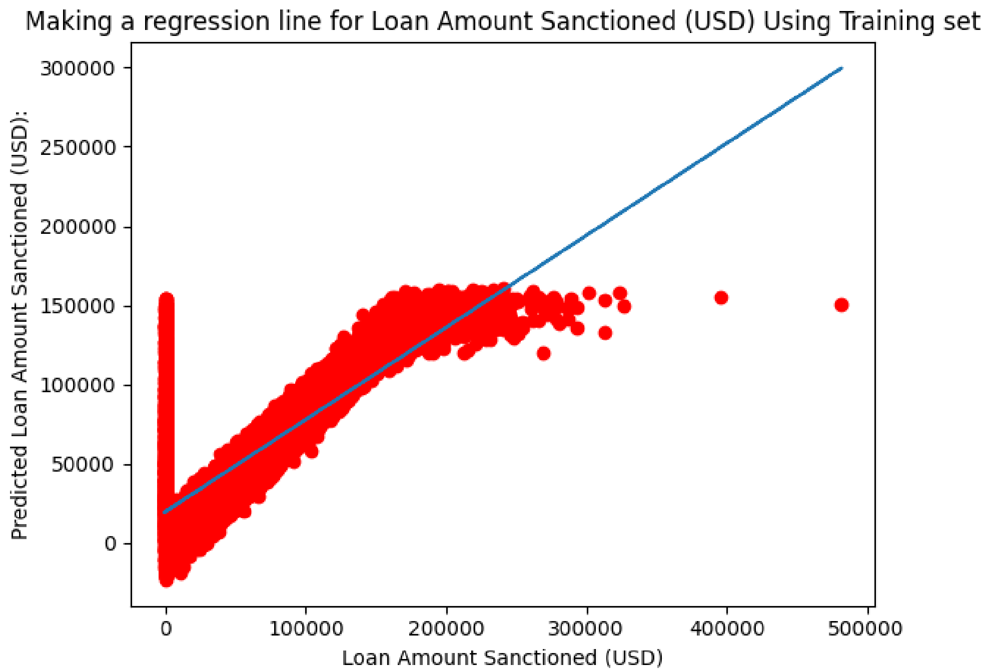

Figure 1: Actual vs Predicted plot in Training set using polynomial features

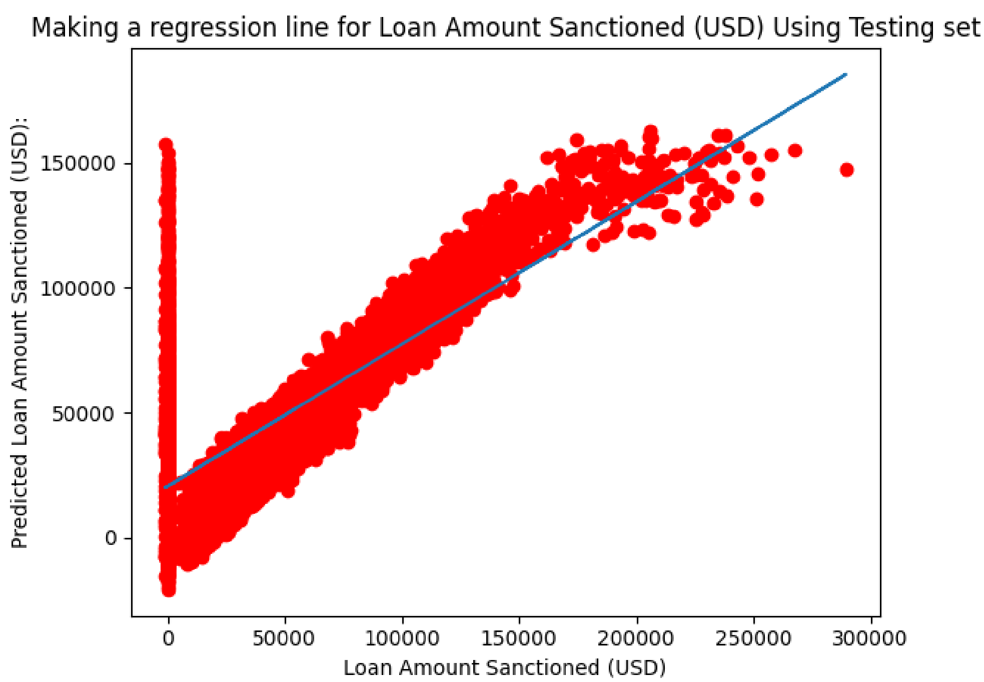

Figure 2: Actual vs Predicted plot in Training set using polynomial features

Results Tables:

Table 1: Cross Validation Results (K=5)

Fold	**MAE**	**MSE**	**RMSE**	R^2 **Score**
1	21557.2096	1013514715.123	31835.745	0.55
2	21579.279	988206623.080	31435.753	0.56
3	21255.7439	973672396.7435	31203.724	0.57
4	21515.650	962460326.0066	31023.544	0.60
5	21751.9826	1032099267.754	32126.301	0.56
Average	**2**1531.9732	**993990665.7415**	**31525.0140**	**0.6038**

Table 2: Summary of Model Performance and Setup

Description	**Result**
Dataset Size (after preprocessing)	29,660
Train/Test Split Ratio	Training set 80%, Testing set 20%
Features used for prediction	14
Model Used	Linear Regression
Cross-Validation Used?	Yes
If yes, no. of folds (k)	5
Reference to CV results table	Table 1
MAE on Test Set	21,557.2096
MSE on Test Set	1013514715.123
RMSE on Test Set	31835.7458
R^2 score on Test Set	0.55347
Adjusted R^2 score on Test Set	0.553247
Most Influential Feature	Loan Amount Request (USD)
Observations from residual plot	There's a vertical cluster around actual values = 0(low loan cases).
Interpretation of Predicted vs Actual Plot	For small values of the target, there is a relatively high number of incorrect predictions.
Any Overfitting or underfitting observed?	No
If yes, brief justification	-

Learning Practices :

- I have Learnt to implement and evaluate Linear Regression.
- I have Interpreted regression metrics and model outputs.
- I have Practiced visualization and reporting of model performance.