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

Degree & Branch	B.E. 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 (Include here): Title (to be included here)

Aim: To 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.

Libraries used:

- pandas
- numpy
- matplotlib.pyplot
- seaborn
- sklearn.linear_model.LinearRegression
- sklearn.model_selection.StratifiedKFold
- sklearn.metrics.mean_absolute_error
- sklearn.metrics.mean_squared_error
- sklearn.metrics.r2_score
- sklearn.preprocessing.LabelEncoder
- sklearn.preprocessing.StandardScaler
- pandas.get_dummies

theoritical description of the algorithm:

Cross-Validation Strategy

To evaluate the model effectively, a 5-fold stratified cross-validation approach was used. This ensures that each fold maintains a balanced distribution of the target variable, leading to a more reliable and unbiased performance estimation.

Data Preparation

The training and testing datasets were combined to facilitate uniform preprocessing and feature engineering. This step allowed for consistent data transformations and feature extraction across the entire dataset.

Feature Engineering

Custom features such as the Loan-to-Income ratio, Total Expenses-to-Income ratio, and Loan-to-Value (LTV) ratio were created. These derived metrics were essential in reflecting the customer's financial condition and credit risk, offering the model additional insights beyond raw data.

Data Cleaning

Columns like Customer ID, Property ID and Name, which had no predictive value, were removed. Missing values in numerical columns were imputed with the mean, while categorical variables were filled with the most frequent value (mode) to preserve data integrity.

Encoding and Scaling

Categorical features were converted into numeric format using Label Encoding. Numerical features were standardized using `StandardScaler`, ensuring they were on a similar scale and improving the model's convergence and performance.

Exploratory Data Analysis

EDA included visualizations such as boxplots and heatmaps. Boxplots helped detect outliers in features like income and loan amount, while heatmaps illustrated correlations between features and the loan amount, guiding better feature selection.

Model Training and Evaluation

A Linear Regression model was trained using the preprocessed data. Model coefficients were analyzed to interpret feature importance. Evaluation included Actual vs Predicted plots and Residual plots, which confirmed that the model's predictions were consistent and generalizable.

Code Implementation:

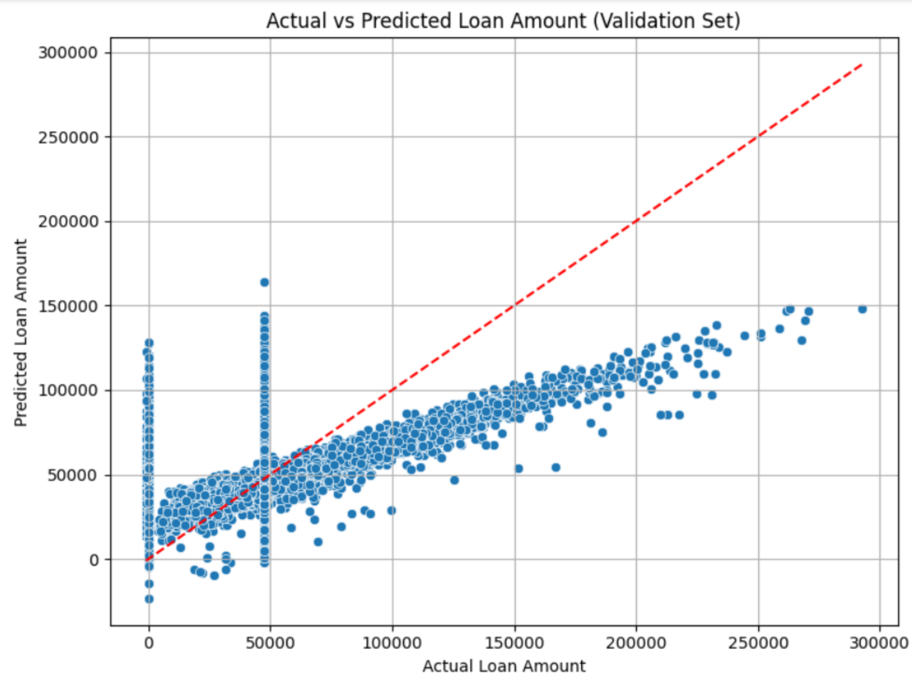
```
# -*- coding: utf-8 -*-
"""ml lab 2.ipynb"""

from google.colab import files
import pandas as pd

uploaded = files.upload()
train_df = pd.read_csv('train.csv')
train_df['source'] = 'train'
train_df.head()

uploaded = files.upload()
```

```
test_df = pd.read_csv('test.csv')
test_df['source'] = 'test'

if 'Loan Sanction Amount (USD)' not in test_df.columns:
    test_df['Loan Sanction Amount (USD)'] = None

df = pd.concat([train_df, test_df], ignore_index=True)

unnecessary_columns = ['Customer ID', 'Name', 'Property ID']
df.drop(columns=unnecessary_columns, inplace=True)

df['Loan Amount Request (USD)'] = pd.to_numeric(df['Loan Amount Request (USD)'], errors='coerc
df['Property Price'] = pd.to_numeric(df['Property Price'], errors='coerce')
df['Income (USD)'] = pd.to_numeric(df['Income (USD)'], errors='coerce')
df['Current Loan Expenses (USD)'] = pd.to_numeric(df['Current Loan Expenses (USD)'], errors='c

def ltv_risk_category(row):
    loan_amount = row['Loan Amount Request (USD)']
    property_price = row['Property Price']
    if pd.isna(loan_amount) or pd.isna(property_price) or property_price == 0:
        return 'Unknown'
    ltv = loan_amount / property_price
    if ltv >= 0.9: return 'Very High'
    elif ltv >= 0.75: return 'High'
    elif ltv >= 0.6: return 'Moderate'
    else: return 'Low'

df['LTV_Risk'] = df.apply(ltv_risk_category, axis=1)
df['Loan_to_Income'] = df['Loan Amount Request (USD)'] / (df['Income (USD)'] + 1e-5)
df['Total_Expenses_to_Income'] = df['Current Loan Expenses (USD)'] / (df['Income (USD)'] + 1e-

# Fill missing values
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
for col in numeric_cols:
    df[col] = df[col].fillna(df[col].mean())

categorical_cols = df.select_dtypes(include=['object']).columns
for col in categorical_cols:
    df[col] = df[col].fillna(df[col].mode()[0])

from sklearn.preprocessing import LabelEncoder
label_encoders = {}
for col in df.select_dtypes(include=['object']).columns:
    df[col] = df[col].astype(str)
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le
```

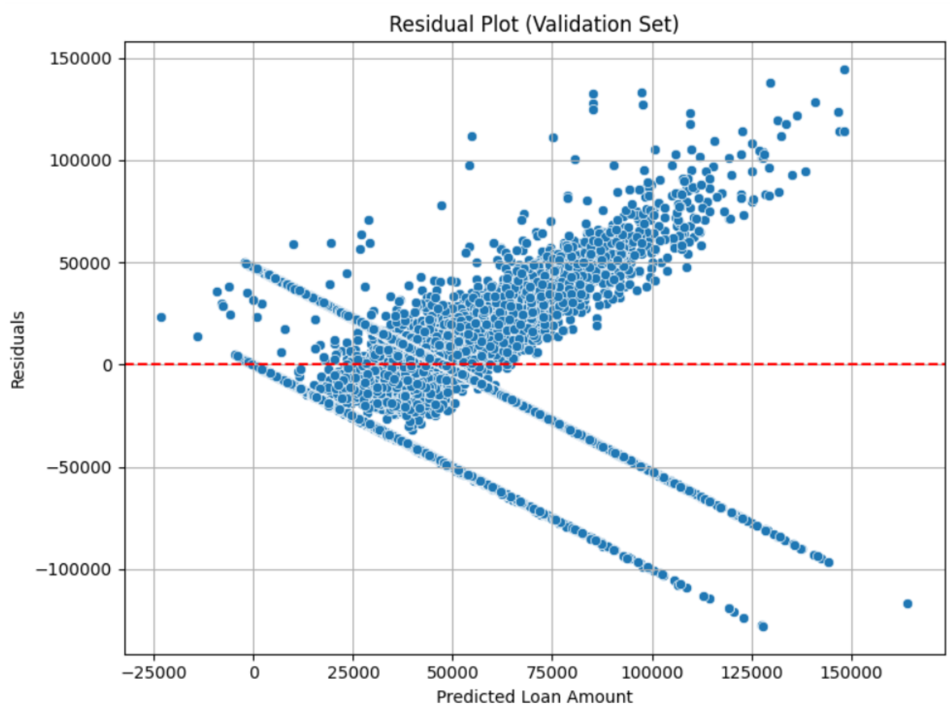
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
features = df.drop('Loan Sanction Amount (USD)', axis=1)
fs = scaler.fit_transform(features)
fs = pd.DataFrame(fs, columns=features.columns)
fs['Loan Sanction Amount (USD)'] = df['Loan Sanction Amount (USD)']

from sklearn.model_selection import StratifiedKFold
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

X = fs.drop('Loan Sanction Amount (USD)', axis=1)
y = fs['Loan Sanction Amount (USD)']
X = pd.get_dummies(X, drop_first=True)
y_binned = pd.qcut(y, q=5, labels=False, duplicates='drop')
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

mae_scores, mse_scores, rmse_scores, r2_scores, adj_r2_scores = [], [], [], [], []
fold = 1
results = []

def adjusted_r2(r2, n, k):
    return 1 - (1 - r2) * (n - 1) / (n - k - 1)

for train_idx, val_idx in skf.split(X, y_binned):
    X_train_fold, X_val_fold = X.iloc[train_idx], X.iloc[val_idx]
    y_train_fold, y_val_fold = y.iloc[train_idx], y.iloc[val_idx]

    model = LinearRegression()
    model.fit(X_train_fold, y_train_fold)
    y_pred = model.predict(X_val_fold)

    mae = mean_absolute_error(y_val_fold, y_pred)
    mse = mean_squared_error(y_val_fold, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_val_fold, y_pred)
    adj_r2 = adjusted_r2(r2, X_val_fold.shape[0], X_val_fold.shape[1])

    mae_scores.append(mae)
    mse_scores.append(mse)
    rmse_scores.append(rmse)
    r2_scores.append(r2)
    adj_r2_scores.append(adj_r2)

    results.append([f"Fold {fold}", mae, mse, rmse, r2, adj_r2])
    fold += 1
```

```
results.append([
    "Average",
    np.mean(mae_scores),
    np.mean(mse_scores),
    np.mean(rmse_scores),
    np.mean(r2_scores),
    np.mean(adj_r2_scores)
])

import pandas as pd
cv_results_df = pd.DataFrame(results, columns=["Fold", "MAE", "MSE", "RMSE", "R² Score", "Adjus
print(cv_results_df)

# -------------------- ACTUAL vs PREDICTED PLOT --------------------
plt.figure(figsize=(8, 6))
sns.scatterplot(x=y_val, y=model.predict(X_val))
plt.xlabel("Actual Loan Amount")
plt.ylabel("Predicted Loan Amount")
plt.title("Actual vs Predicted Loan Amount (Validation Set)")
plt.plot([y_val.min(), y_val.max()], [y_val.min(), y_val.max()], color='red', linestyle='--')
plt.grid(True)
plt.tight_layout()
plt.show()

# -------------------- RESIDUAL PLOT --------------------
residuals = y_val - model.predict(X_val)

plt.figure(figsize=(8, 6))
sns.scatterplot(x=model.predict(X_val), y=residuals)
plt.axhline(0, color='red', linestyle='--')
plt.xlabel("Predicted Loan Amount")
plt.ylabel("Residuals")
plt.title("Residual Plot (Validation Set)")
plt.grid(True)
plt.tight_layout()
plt.show()
```

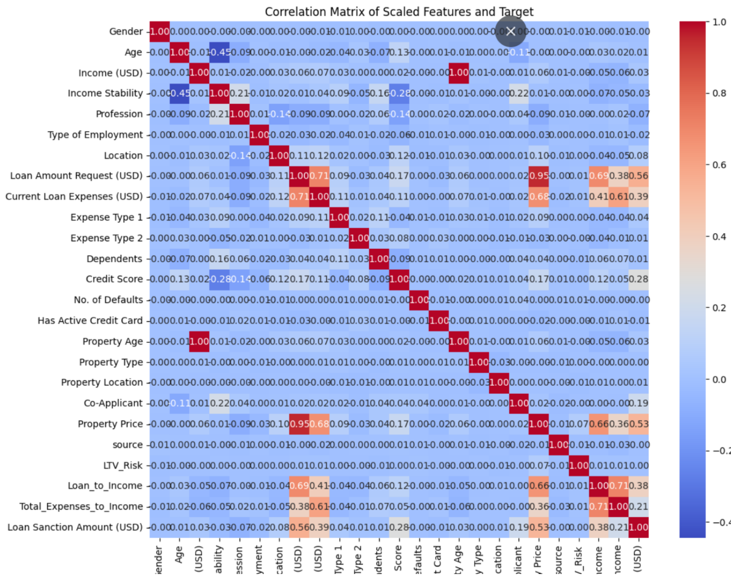
Screenshots of Output:

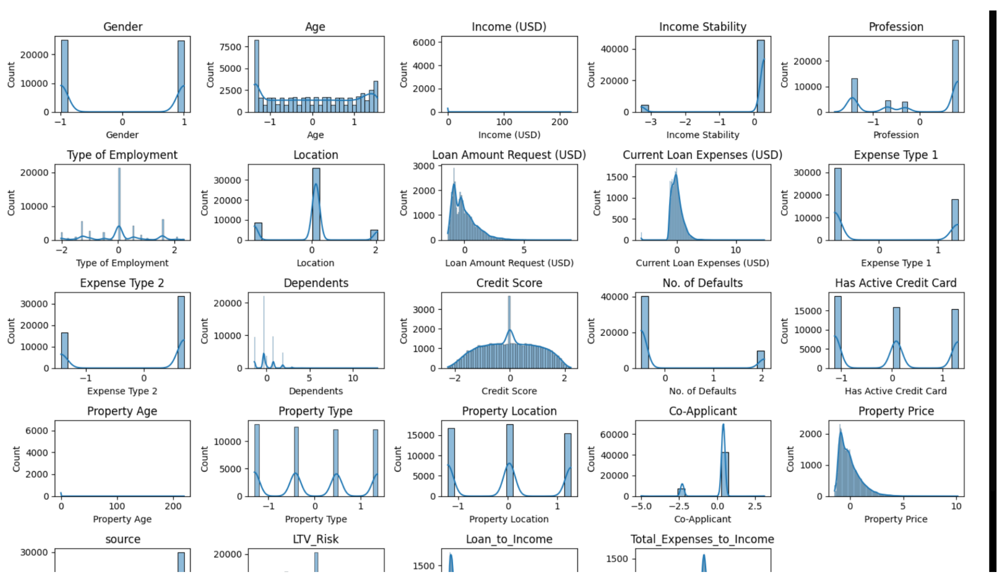


Actual vs Predicted Loan Amount (Validation Set)
Predicted Loan Amount
Actual Loan Amount
0
50000
100000
150000
200000
250000
300000



Residual Plot (Validation Set)
Residuals
Predicted Loan Amount
150000
100000
50000
0
−50000
−100000
−25000
25000
75000
125000





Result and Discussions:

Fold	MAE	MSE	RMSE	R^2 Score	Adjusted R^2 Score
Fold 1	21696.97	8.35×10^8	28892.30	0.3702	0.3687
Fold 2	22129.75	8.94×10^8	29901.59	0.3692	0.3677
Fold 3	21765.02	8.54×10^8	29231.61	0.3766	0.3751
Fold 4	22145.82	9.09×10^8	30147.18	0.3628	0.3613
Fold 5	21750.26	8.56×10^8	29260.96	0.3688	0.3673
Average	**21897.56**	8.70×10^8	**29486.73**	**0.3695**	**0.3680**

Table 1: Cross-validation metrics across 5 folds including Adjusted R^2 Score

Table 2: Summary of Results for Loan Amount Prediction

Description	**Student's Result**
Dataset Size (after preprocessing)	11819 rows $\times$ 17 columns
Train/Test Split Ratio	5-Fold Stratified Cross-Validation
Feature(s) Used for Prediction	All encoded and scaled features (excluding target)
Model Used	Linear Regression
Cross-Validation Used? (Yes/No)	Yes
If Yes, Number of Folds	5
Reference to CV Results Table	Table 1
Mean Absolute Error (MAE) on Test Set	21,897.56 USD
Mean Squared Error (MSE) on Test Set	8.70×10^8 USD^2
Root Mean Squared Error (RMSE) on Test Set	29,486.73 USD
R^2 Score on Test Set	0.3695
Adjusted R^2 Score on Test Set	0.3680
Most Influential Feature(s)	`Loan Amount Request (USD)`, `Income (USD)`, `Loan_to_Income` (strongest correlation with target)
Observations from Residual Plot	Funnel shape (heteroscedasticity), increasing spread at higher values
Interpretation of Predicted vs Actual Plot	Positive trend with deviations at high amounts; underestimation of large loan values
Any Overfitting or Underfitting Observed?	Mild underfitting
If Yes, Brief Justification	Lower R^2 ($\sim$0.37), increasing residual spread, model underestimates higher targets

Performance Analysis

The model's performance was assessed using Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and both R^2 and Adjusted R^2 scores across five folds of cross-validation. The following observations were made:

- The average MAE of approximately 21,897 indicates the model's typical prediction error in USD.
- The MSE values, ranging around 8.5×10^8 to 9.1×10^8, reflect some variance in prediction errors across folds, with larger errors penalized more heavily.
- RMSE, which also penalizes large errors, is consistently around 29,000, reinforcing the MAE insight with a slightly higher sensitivity to outliers.
- The R^2 score averages to 0.3695, suggesting that roughly 37% of the variance in the target variable is explained by the model. While not extremely high, it reflects moderate predictive power, which may be improved with more advanced feature engineering or model tuning.
- Adjusted R^2 scores are slightly lower than R^2, as expected, accounting for the number of predictors used. The values are fairly consistent across all folds, indicating stable performance.

Learning Practices:

- Learned how to apply Linear Regression for predicting continuous values like loan amounts.
- Understood how cross-validation works and how it helps improve the reliability of model results.
- Practiced cleaning and preparing data, including handling missing values and scaling features.
- Learned how to evaluate models using metrics like MAE, RMSE, and R^2 score.