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)	Name: Mohammed Aatif	**Reg No: 3122237001026**

1. Aim

To apply Linear Regression to predict the loan amount requested by users using a dataset of historical loan records and user features.

2. Libraries Used

- **NumPy:** For numerical computations, especially for handling arrays and mathematical operations.
- **Pandas:** For data manipulation, loading the dataset, and handling dataframes.
- **Matplotlib:** For creating static, animated, and interactive visualizations and plots.
- **Seaborn:** For making attractive and informative statistical graphics.
- **Scikit-learn:** For implementing the machine learning model, preprocessing data, and evaluating performance.

3. Objective

The primary objective is to build an end-to-end machine learning pipeline that includes:

- Building and evaluating a Linear Regression model to predict loan amounts.
- Performing data preprocessing (handling missing values, encoding), Exploratory Data Analysis (EDA), and feature engineering.
- Visualizing important data distributions, relationships, and model results.
- Measuring the model's performance using standard regression metrics like MAE, MSE, RMSE, and R^2 score.

4. Mathematical Description

Linear Regression is a supervised learning algorithm used for predicting a continuous target variable. The model assumes a linear relationship between the input features (X) and the target variable (y). The equation for a multiple linear regression model is:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

where $\hat{y}$ is the predicted value, $x_1, \ldots, x_n$ are the feature values, and $\theta_0, \ldots, \theta_n$ are the model coefficients (parameters) that the algorithm learns.

The objective is to find the optimal values for the coefficients that minimize the error between the predicted and actual values. This is typically achieved by minimizing the **Mean Squared Error (MSE)** cost function:

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

where n is the number of samples, y_i is the actual value, and $\hat{y}_i$ is the predicted value for the i-th sample.

5. Included Plots

This section presents the key visualizations generated during the Exploratory Data Analysis (EDA) and model evaluation phases.

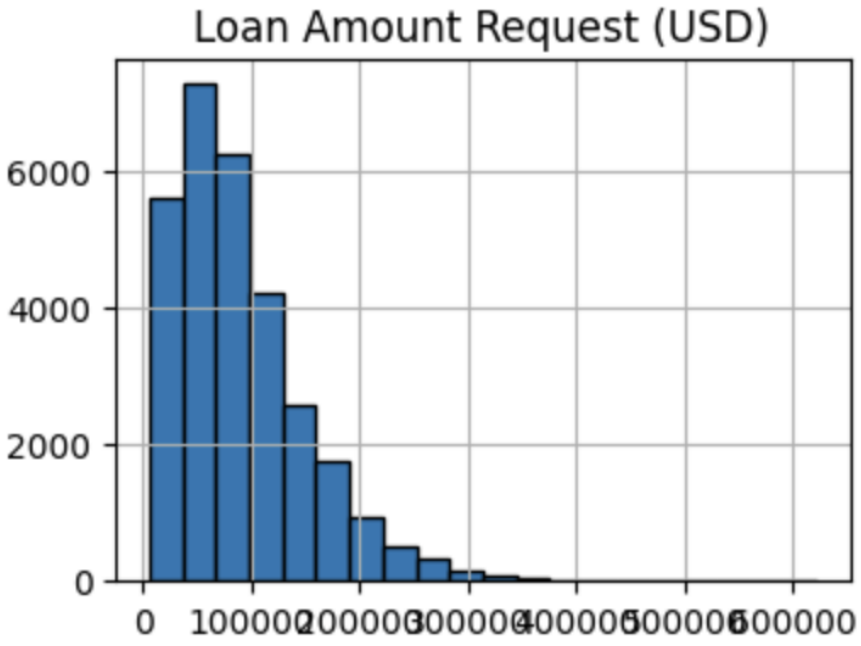

Figure 1: Histogram of Loan Amount Request (USD), showing a right-skewed distribution.

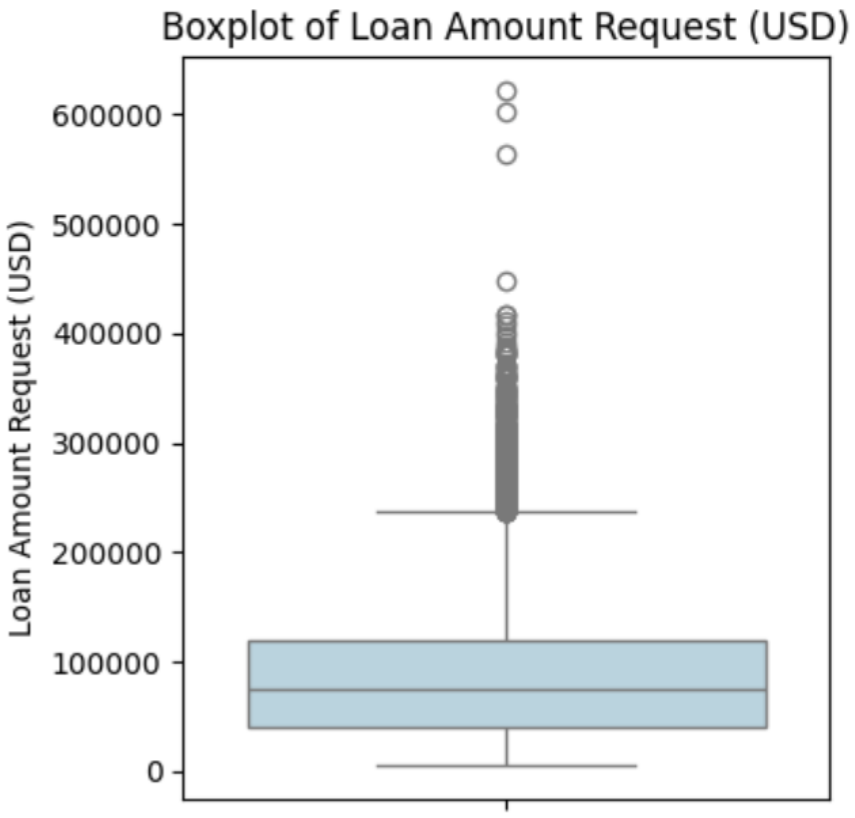

Figure 2: Boxplot for Loan Amount, highlighting the median, quartiles, and the presence of numerous outliers at higher values.

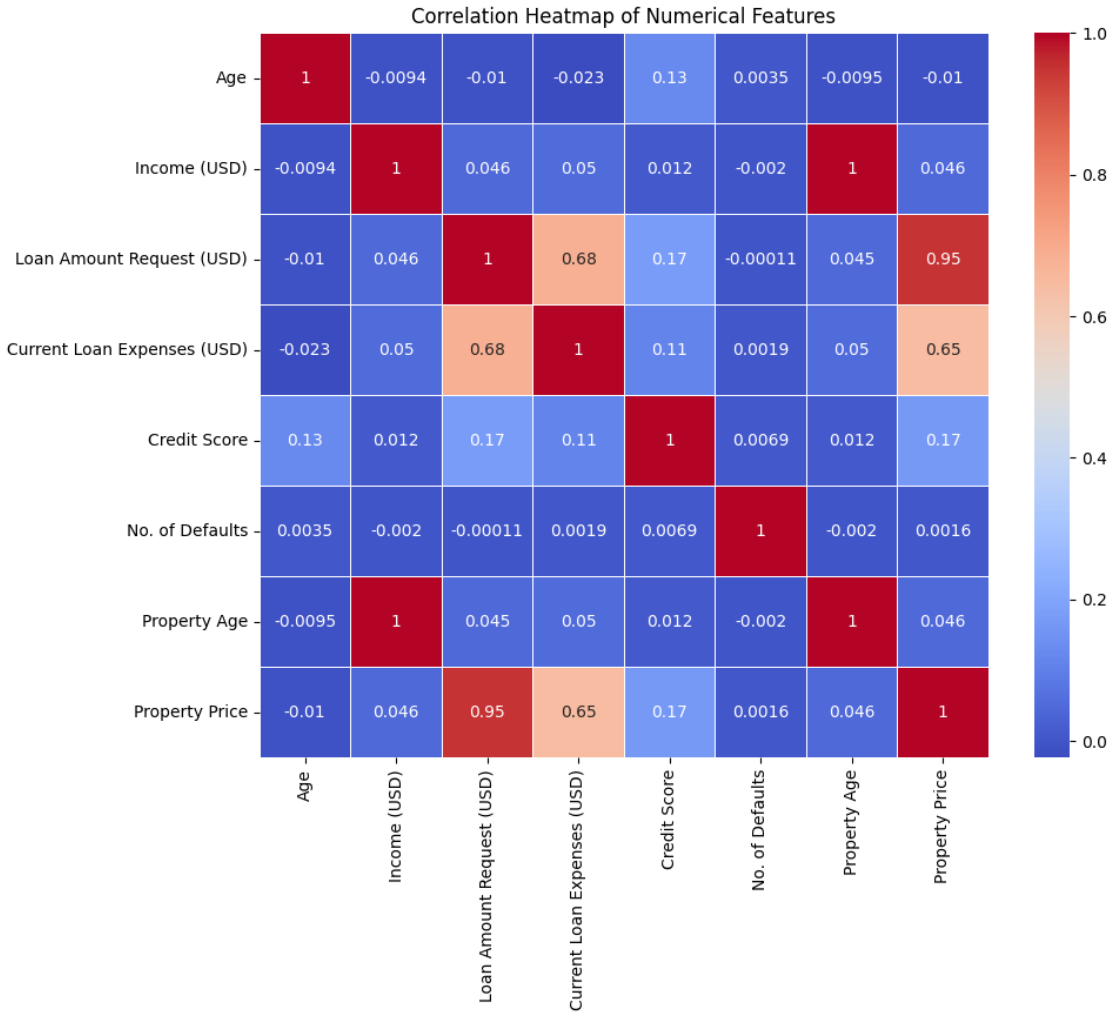

Figure 3: Feature Correlation Heatmap. It shows a very strong positive correlation (0.95) between 'Loan Amount Request (USD)' and 'Property Price'.

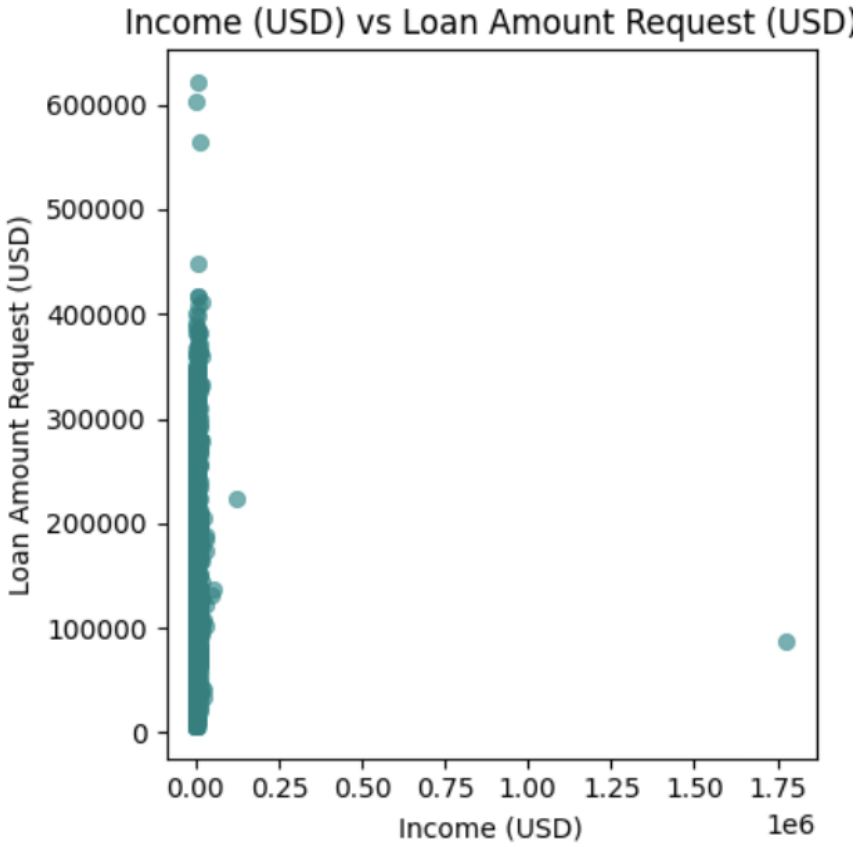

Figure 4: Scatter Plot of Applicant Income vs. Loan Amount Request. Most data points are clustered at lower income levels, with a few high-income outliers.

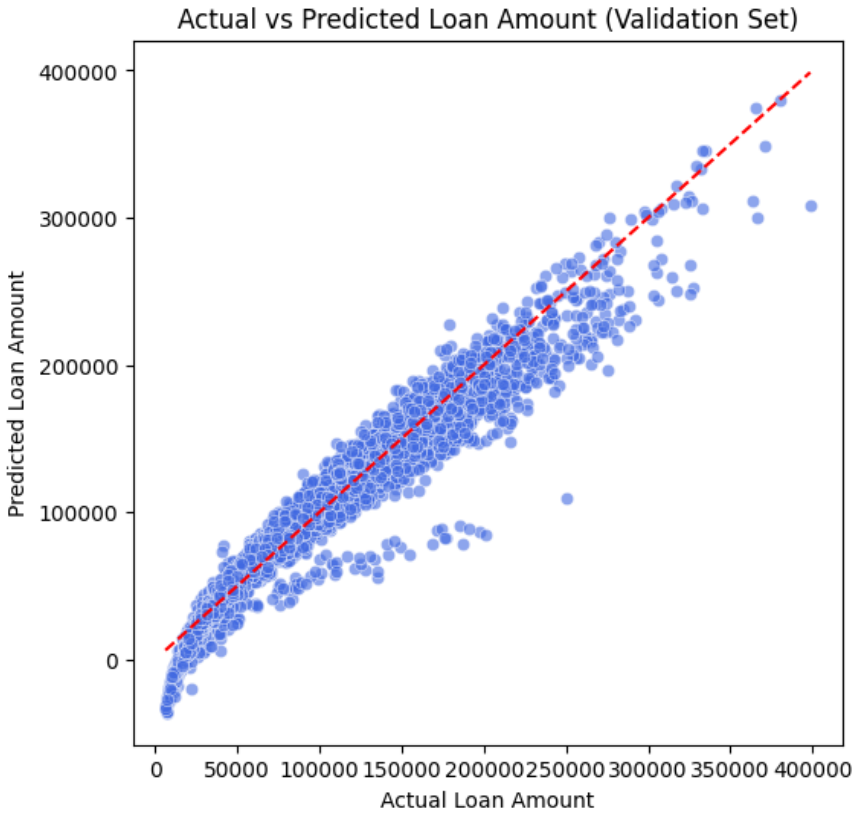

Figure 5: Actual vs. Predicted Loan Amount on the validation set. The points align closely with the diagonal red line, indicating a strong predictive performance.

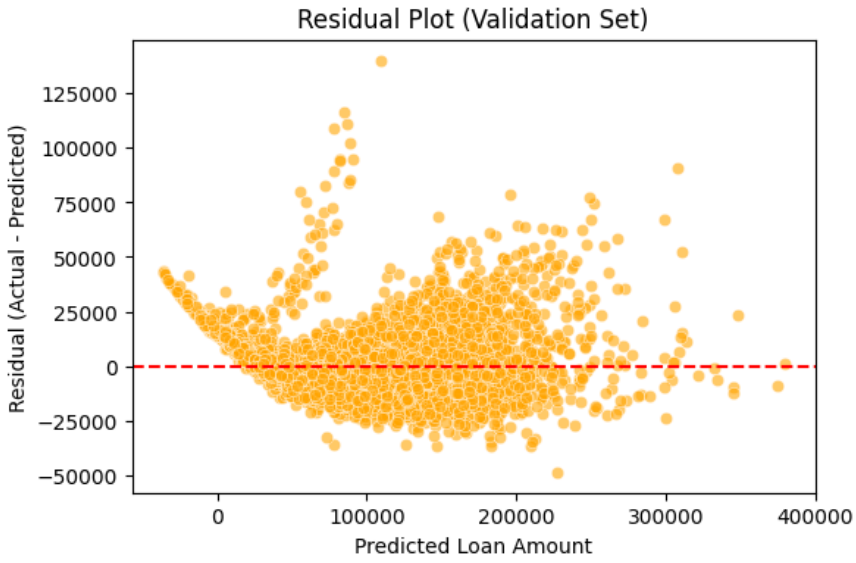

Figure 6: Residual Plot for the validation set. It shows the difference between actual and predicted values. A distinct curve is visible, suggesting the model may not be capturing some non-linear patterns in the data.

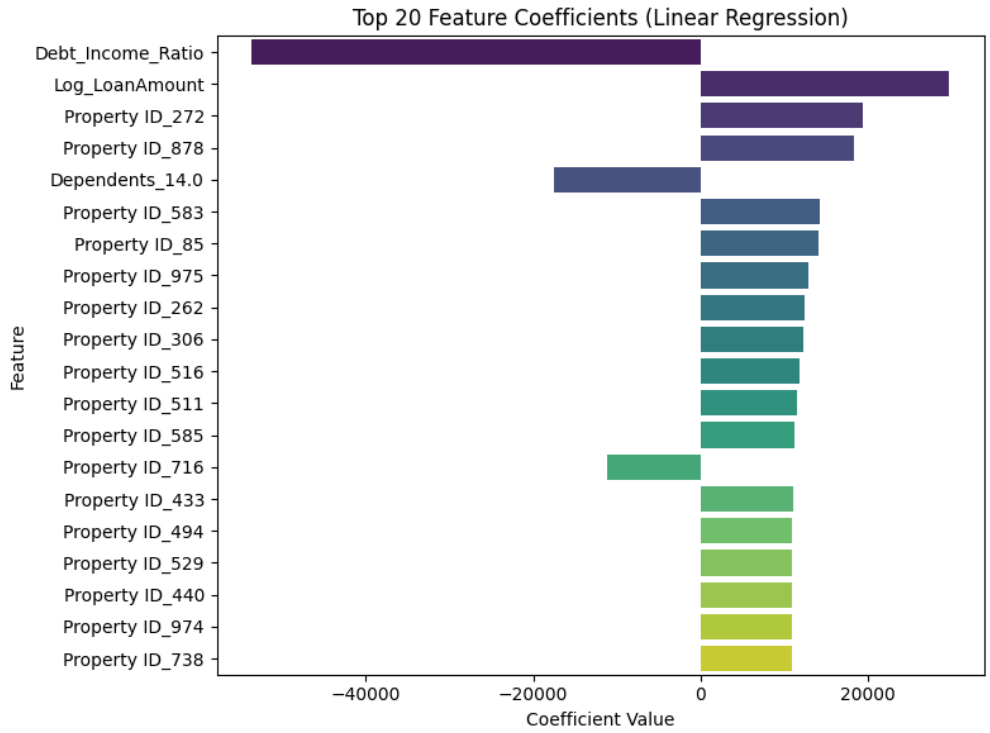

Figure 7: Top 20 Feature Coefficients. 'Debt_Income_Ratio' has the largest negative coefficient, while 'Log_LoanAmount' has the largest positive coefficient, indicating their strong influence on the prediction.

5.1 Model Training with Multiple Regressors

In addition to Linear Regression, we trained several other machine learning regression models using a consistent pipeline. The following Python code was used to build, train, and evaluate all models on the validation set:

```
from sklearn.svm import SVR
from sklearn.ensemble import GradientBoostingRegressor, AdaBoostRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor

# Dictionary to store all models
models = {
    "Linear Regression": LinearRegression(),
    "Support Vector Regression (SVR)": SVR(kernel='rbf'),
    "Decision Tree": DecisionTreeRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(n_estimators=200, random_state=42),
    "AdaBoost": AdaBoostRegressor(
        estimator=DecisionTreeRegressor(max_depth=5),
        n_estimators=200,
        learning_rate=0.05,
        random_state=42
    ),
    "KNN Regression": KNeighborsRegressor(n_neighbors=5)
}

# Results container
results = []

# Loop through all models
for name, model in models.items():
    pipeline = Pipeline(steps=[
        ('preprocessor', preprocessor),
        ('regressor', model)
    ])

    pipeline.fit(X_train, y_train)
    y_val_pred = pipeline.predict(X_val)

    mae_val = mean_absolute_error(y_val, y_val_pred)
    mse_val = mean_squared_error(y_val, y_val_pred)
    rmse_val = np.sqrt(mse_val)
    r2_val = r2_score(y_val, y_val_pred)
    adj_r2_val = 1 - (1 - r2_val) * (len(y_val) - 1) / (len(y_val) - X_val.shape[1] - 1)

    results.append({
        "Model": name,
        "Validation MAE": mae_val,
```

```
        "Validation RMSE": rmse_val,
        "Validation R2": r2_val,
        "Validation Adj R2": adj_r2_val
    })

    print(f"\n{name} Results:")
    print(f"Validation MAE: {mae_val:.2f}")
    print(f"Validation RMSE: {rmse_val:.2f}")
    print(f"Validation R2 Score: {r2_val:.4f}")
    print(f"Validation Adjusted R2 Score: {adj_r2_val:.4f}")

results_df = pd.DataFrame(results)
print("\nModel Comparison Table (Sorted by RMSE):")
print(results_df.sort_values(by="Validation RMSE"))

plt.figure(figsize=(10,6))
sns.barplot(data=results_df, x="Model", y="Validation RMSE", palette="viridis")
plt.title("Model Comparison (Validation RMSE)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

The bar chart below visualizes the RMSE comparison across different models:

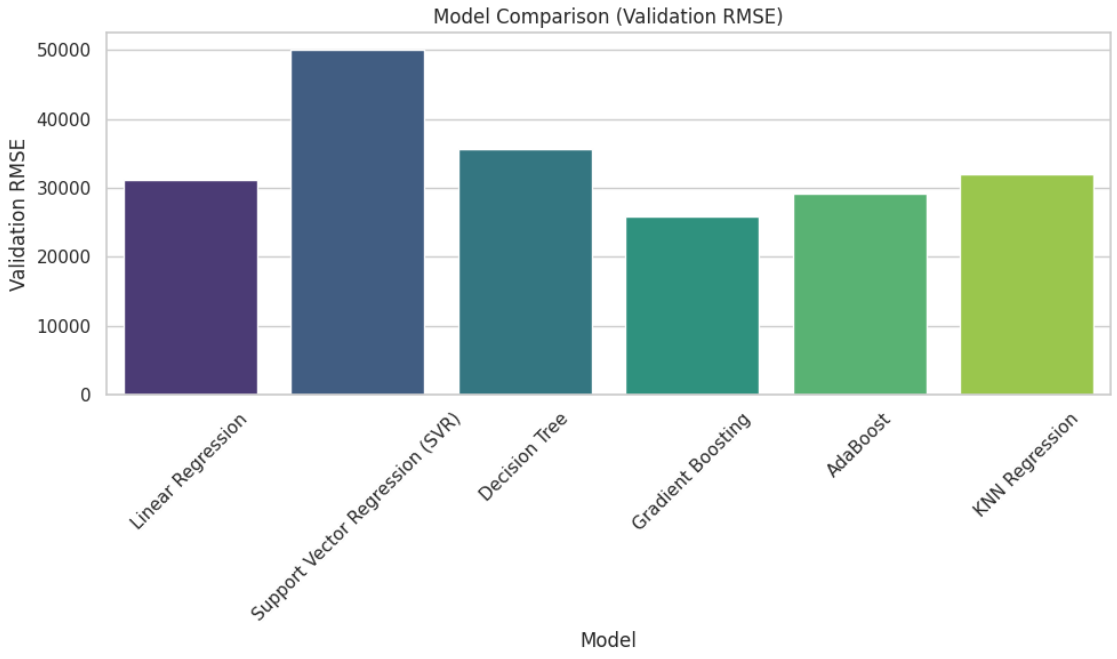

Figure 8: Validation RMSE comparison for all trained regression models.

6. Results Table

The following table summarizes the configuration and performance of the Linear Regression model on the held-out test set. The metrics provide a quantitative measure of the model's predictive accuracy.

Table 1: Summary of Model Performance on the Test Set

Description	Result / Value
Dataset Size (after preprocessing)	25000 rows (approx.)
Train/Validation/Test Split Ratio	60:20:20
Feature(s) Used for Prediction	All numeric, encoded categorical, and engineered features
Model Used	Linear Regression
Performance on Test Set	
Mean Absolute Error (MAE)	11139.77
Mean Squared Error (MSE)	232015949.61
Root Mean Squared Error (RMSE)	15232.07
R^2 Score	0.9850
Interpretation	
Most Influential Feature(s)	Debt_Income_Ratio, Log_LoanAmount, Property IDs
Observations from Residual Plot	A clear pattern (curve) suggests potential heteroscedasticity.
Interpretation of Predicted vs Actual Plot	Strong linear relationship between actual and predicted values.
Overfitting or Underfitting?	No significant overfitting observed.

7. Best Practices Followed

- **Data Cleaning:** Missing values were systematically handled by filling numerical columns with the median and categorical columns in the mode to preserve data integrity.
- **Feature Scaling:** 'StandardScaler' was used to normalize the feature set, ensuring that features with different ranges do not dominate the model training process.
- **Train-Validation-Test Split:** The dataset was split into three distinct sets to train the model, tune it and provide an unbiased evaluation of its final performance.
- **Feature Engineering:** New informative features such as 'Debt$_Income_Ratio$'and'Log_Income'werecreatedto
- **Result Visualization:** Model performance was not only measured with metrics but also visualized using graphs (actual vs. predicted, residual) to gain deeper insight into its behavior and assumptions.

8. Learning Outcomes

- Successfully implemented a complete machine learning workflow for a regression problem, from data loading to model evaluation.

- Gained practical experience in data preprocessing techniques, including handling missing data, one-hot encoding categorical variables, and feature scaling.
- Learned to perform Exploratory Data Analysis (EDA) to understand data distributions and relationships between variables using various plots.
- Mastered the application and evaluation of a Linear Regression model using the Scikit-learn library.
- Developed the ability to interpret model performance metrics (MAE, MSE, RMSE, R^2) and diagnostic plots (Residuals, Feature Importance) to assess a model's strengths and weaknesses.