Experiment 2: Loan Amount Prediction using Linear Regression

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

Name: SPINOLA THERES N
Roll Number: 3122237001051

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

1 Aim

To develop and evaluate a Linear Regression model that predicts the loan sanction amount using historical loan data and borrower features, and to visualize and interpret the results to gain insights into model performance.

2 Libraries Used

- **Pandas** – for data manipulation and analysis
- **NumPy** – for numerical operations and array handling
- **Scikit-learn** – for machine learning model building, preprocessing, and evaluation
- **Matplotlib** – for data visualization and plotting
- **Seaborn** – for statistical data visualization
- **Warnings** – for filtering warning messages
- **Pathlib** – for file path handling

3 Objective

- Load and preprocess the loan dataset from Kaggle
- Handle missing values and encode categorical variables
- Perform exploratory data analysis (EDA) to understand data distributions
- Apply feature scaling and engineering techniques
- Split the dataset into training, validation, and testing sets
- Train and validate Linear Regression and SVR models
- Evaluate model performance using multiple metrics (MAE, MSE, RMSE, R^2)
- Perform K-fold cross-validation for robust model evaluation
- Visualize results through various plots and interpret findings
- Compare model performances and identify the best approach

4 Mathematical Description

The Linear Regression model is mathematically represented as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon$$

Where:

- y = Loan Sanction Amount (USD) - target variable
- $x_1, x_2, ..., x_n$ = input features (age, income, credit score, etc.)
- β_0 = intercept term
- $\beta_1, \beta_2, ..., \beta_n$ = feature coefficients
- ϵ = error term

The model parameters are estimated using the Normal Equation:

$$\boldsymbol{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Evaluation metrics used:

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

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

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{3}$$

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4}$$

5 Code Implementation

5.1 Data Loading and Preprocessing

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.linear_model import LinearRegression

# Load dataset
df = pd.read_csv('train.csv')
print(f"Dataset shape: {df.shape}")

# Drop non-predictive columns
df.drop(columns=['Customer ID', 'Name', 'Property ID'], inplace=True)

# Handle missing values
for col in df.select_dtypes(include='object').columns:
    if df[col].isnull().sum() > 0:
        df[col].fillna(df[col].mode(dropna=True)[0], inplace=True)

for col in df.select_dtypes(include=[np.number]).columns:
    if df[col].isnull().sum() > 0:
        df[col].fillna(df[col].mean(), inplace=True)
```

5.2 Feature Encoding and Scaling

```
# Target and features separation
target_col = 'Loan Sanction Amount (USD)'
y = df[target_col]
X = df.drop(columns=[target_col])

# Encode categorical variables
label_encoder = LabelEncoder()
obj_cols = X.select_dtypes(include='object').columns.tolist()
binary_cols = [col for col in obj_cols if X[col].nunique(dropna=True) == 2]

for col in binary_cols:
    X[col] = label_encoder.fit_transform(X[col].astype(str))

multi_cols = [col for col in obj_cols if col not in binary_cols]
if multi_cols:
    X = pd.get_dummies(X, columns=multi_cols, drop_first=True)

# Feature scaling
scaler = StandardScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

5.3 Model Training and Evaluation

```
# Train-validation-test split
X_train, X_temp, y_train, y_temp = train_test_split(
    X_scaled, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.5, random_state=42)

# Train Linear Regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# Evaluate model
def evaluate_model(model, X_data, y_true, name="Set"):
    y_pred = model.predict(X_data)
    mae = mean_absolute_error(y_true, y_pred)
    mse = mean_squared_error(y_true, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_true, y_pred)
    return mae, mse, rmse, r2

lr_test_results = evaluate_model(lr_model, X_test, y_test, "Test")
```

6 Included Plots

The following visualizations were generated as part of the analysis:

- **Distribution Plots:** Histograms showing the distribution of loan amounts and key numerical features
- **Scatter Plots:** Examining relationships between income, credit score, and loan amounts
- **Correlation Heatmap:** Identifying multicollinearity and feature relationships
- **Boxplots:** Detecting outliers in numerical features
- **Actual vs Predicted Plot:** Visual evaluation of model performance
- **Residual Plot:** Assessment of linearity assumptions and residual distribution
- **Feature Coefficients Bar Plot:** Interpretation of feature importance in the linear model
- **Model Comparison Plots:** Performance comparison between Linear Regression and SVR

6.1 Plot Placeholders

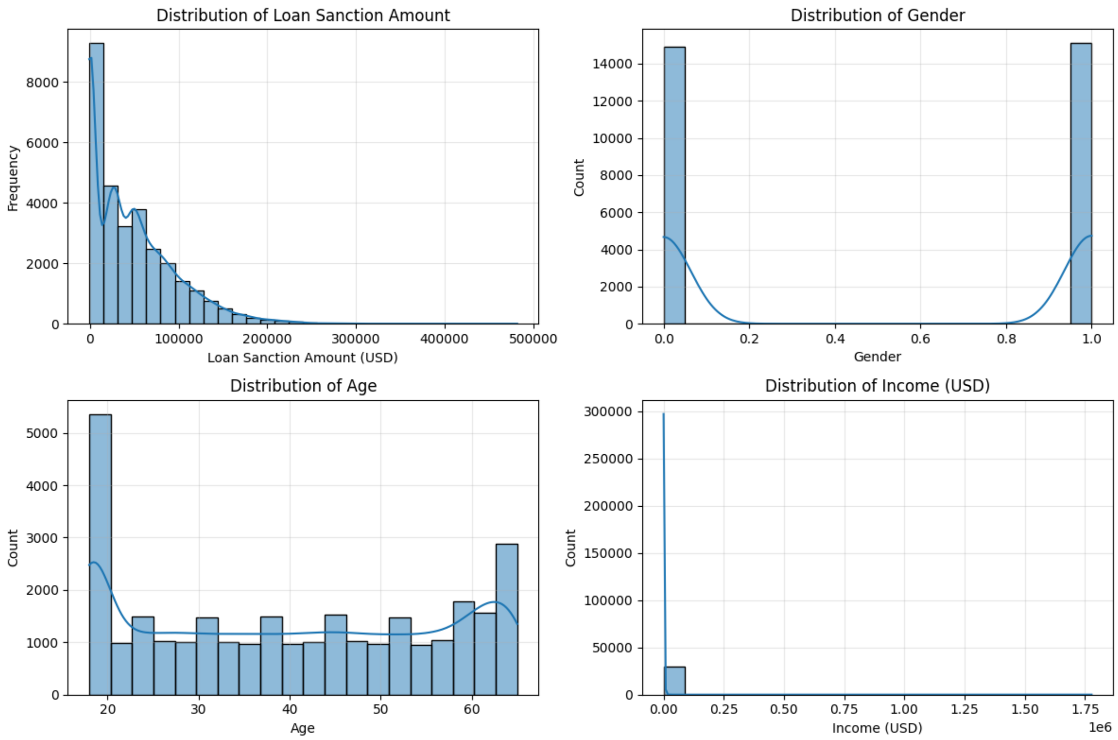

Figure 1: Distribution of Loan Sanction Amount and Key Features

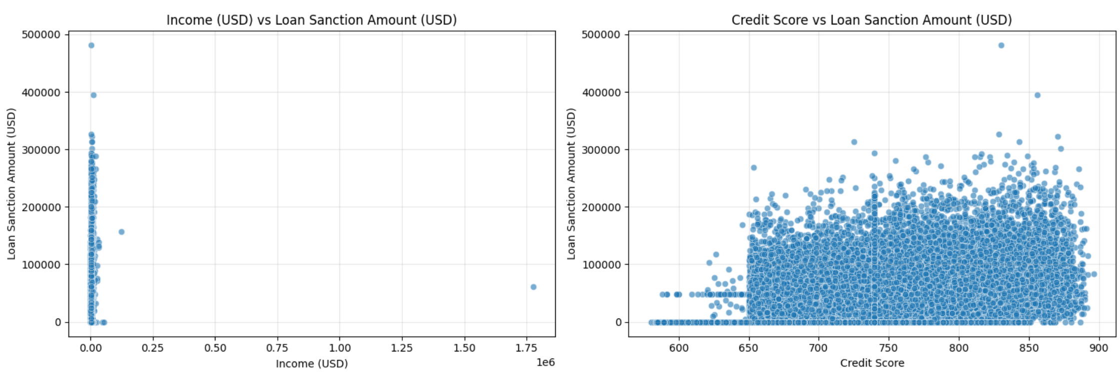

Figure 2: Scatter Plots: Key Features vs Loan Amount

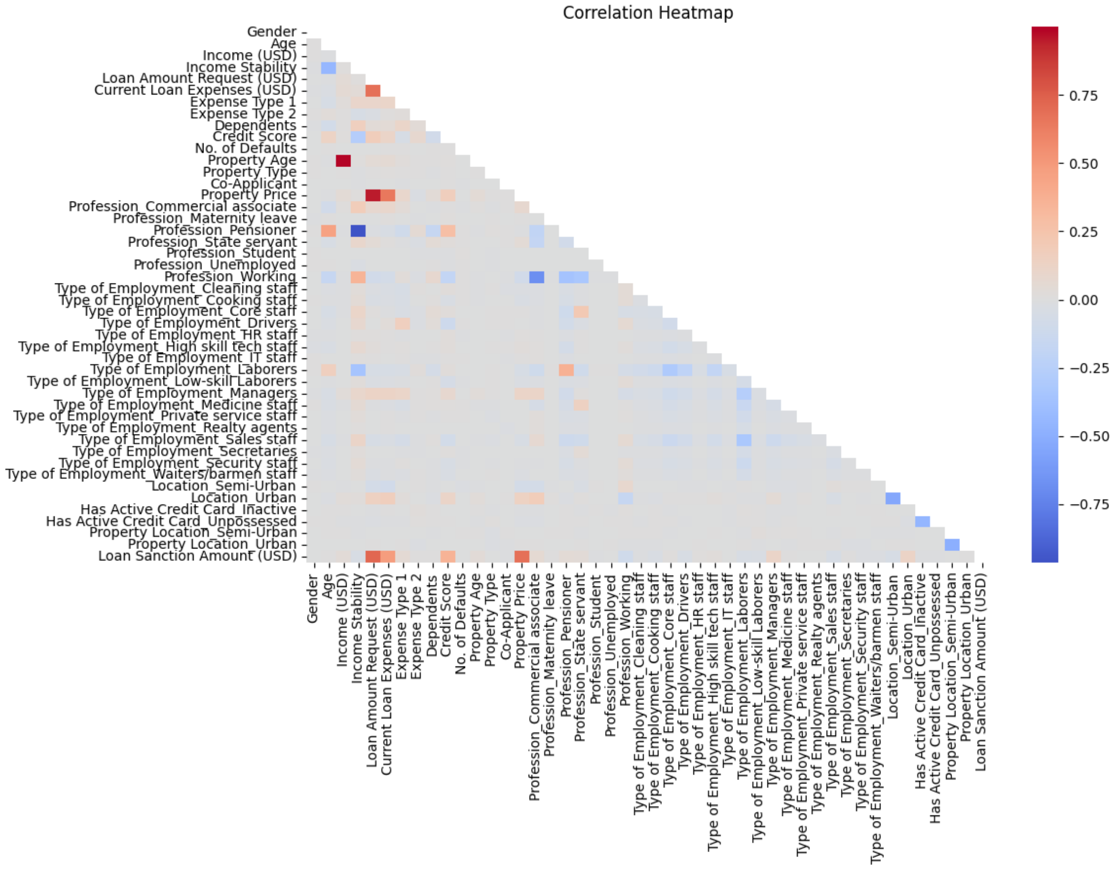

Figure 3: Correlation Heatmap

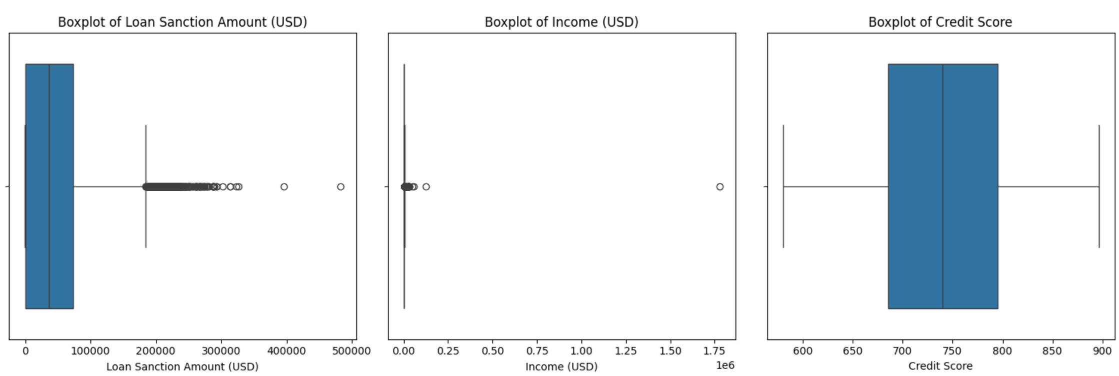

Figure 4: Boxplots for Outlier Detection

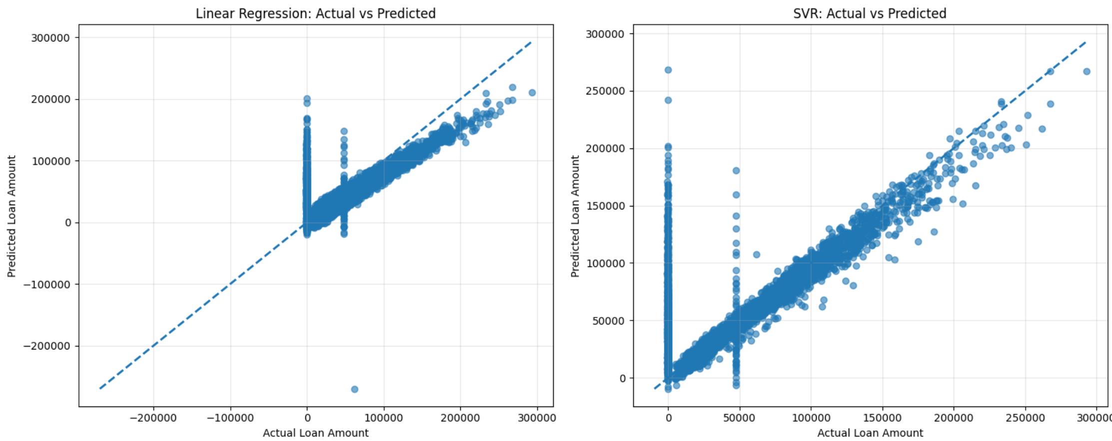

Figure 5: Actual vs Predicted Values Comparison

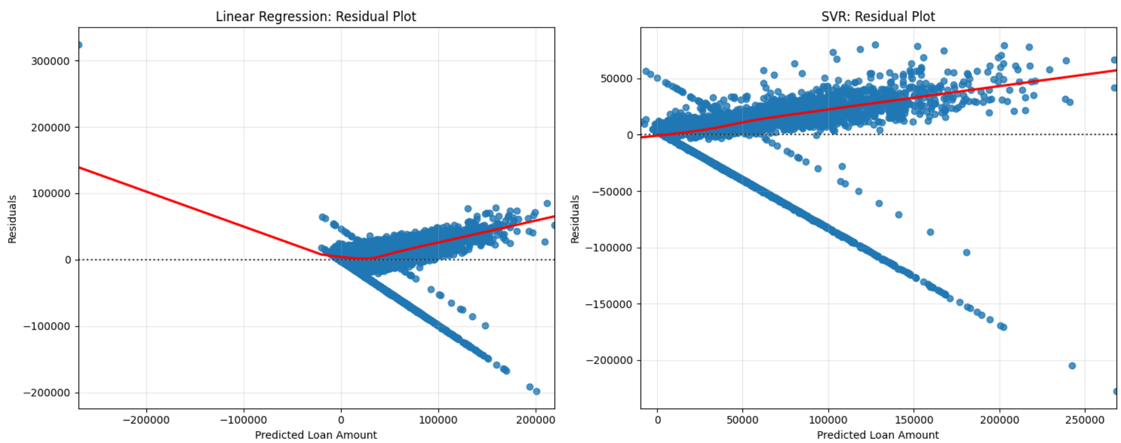

Figure 6: Residual Plots

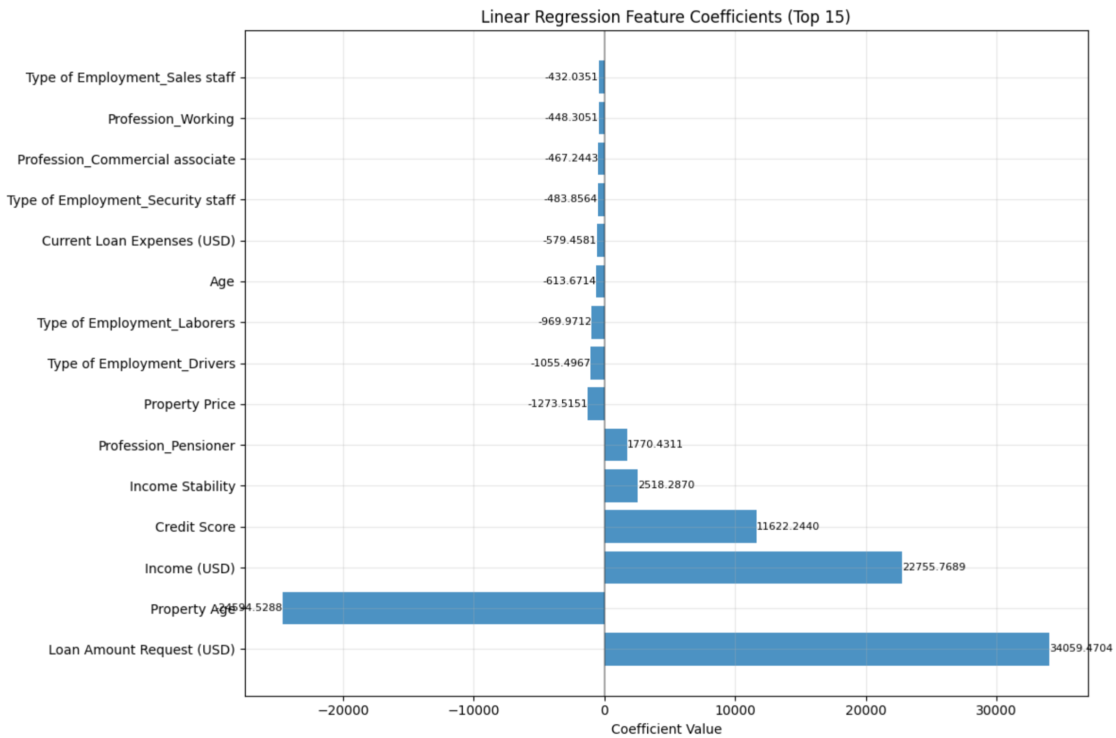

Figure 7: Linear Regression Feature Coefficients

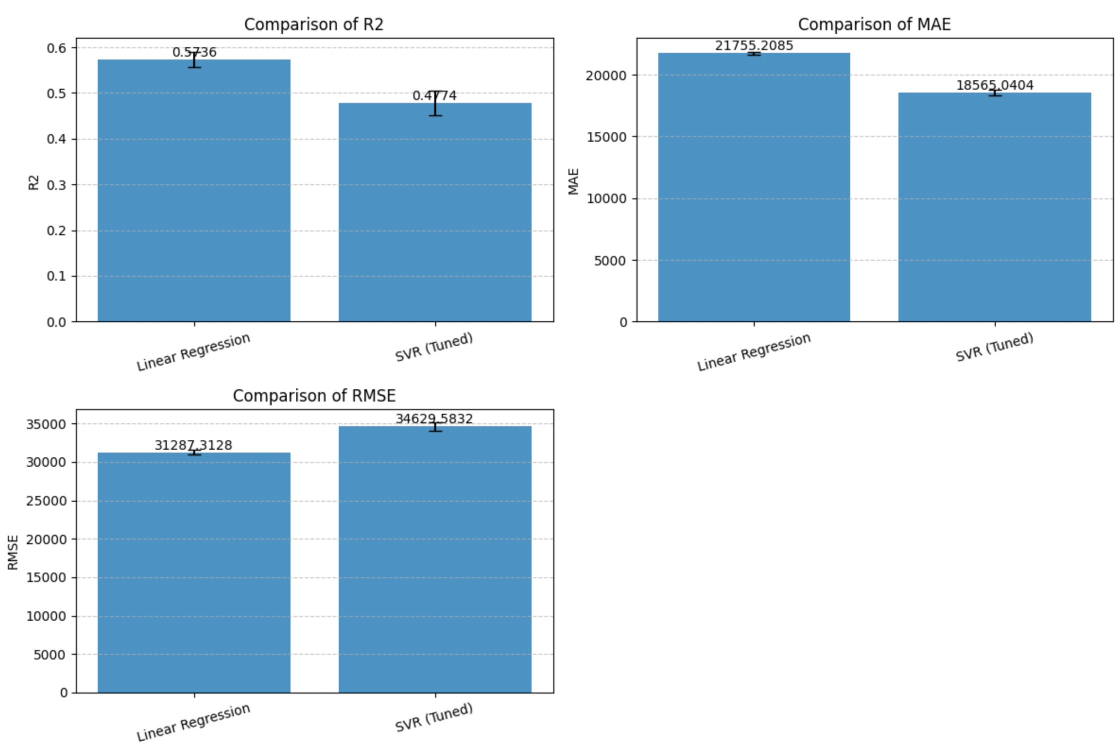

Figure 8: Model Performance Comparison

7 Results Tables

7.1 Cross-Validation Results

Table 1: Cross-Validation Results (K = 5)

Fold	**MAE**	**MSE**	**RMSE**	**R^2 Score**
Fold 1	21751.49	1000187741.75	31625.74	0.5649
Fold 2	21853.81	979222213.16	31292.53	0.5690
Fold 3	22386.75	1064073204.66	32620.14	0.5403
Fold 4	21760.06	995920512.76	31558.21	0.5764
Fold 5	21023.49	880308985.94	29670.00	0.6100
Average	**21755.12**	**983942531.65**	**31353.32**	**0.5721**

7.2 Model Performance Comparison

Table 2: Model Comparison (Cross-Validation Results)

Model	**R^2 Mean**	**R^2 Std**	**RMSE Mean**	**RMSE Std**	**MAE Mean**	**MAE Std**
Linear Regression	0.5736	0.0169	31287.31	364.92	21755.21	121.05
SVR (Tuned)	0.4774	0.0270	34629.58	520.47	18565.04	223.83

7.3 Summary of Results for Loan Amount Prediction

Table 3: Summary of Results for Loan Amount Prediction

gray!20 **Description**	**Linear Regression Result**	**SVR Result**
Dataset Size (after preprocessing)	30000 samples, 45 features	30000 samples, 45 features
Train/Test Split Ratio	70% Train, 15% Validation, 15% Test	70% Train, 15% Validation, 15% Test
Features Used for Prediction	45 features (scaled)	45 features (scaled)
Model Used	Linear Regression	SVR (linear kernel)
Cross-Validation Used? (Yes/No)	Yes	Yes (for hyperparameter tuning)
Number of Folds (K)	5	3
Reference to CV Results Table	Table 1	Hyperparameter tuning results
blue!10 Mean Absolute Error (MAE) on Test Set	21967.53	18422.17
blue!10 Mean Squared Error (MSE) on Test Set	992766452.72	1188521269.15
blue!10 Root Mean Squared Error (RMSE) on Test Set	31508.20	34474.94
green!10 R^2 Score on Test Set	0.5595	0.4726
green!10 Adjusted R^2 Score on Test Set	0.5550	0.4673
Most Influential Feature(s)	Loan Amount Request (USD), Property Age	Feature importance not available for SVR
Observations from Residual Plot	Mean: 306.22, Std: 31506.71	Mean: -11354.70, Std: 32551.38
Interpretation of Predicted vs Actual Plot	Model explains 55.9% of variance	Model explains 47.3% of variance
Any Overfitting or Underfitting Observed?	No	N/A
Brief Justification	Train R^2: 0.5783, Test R^2: 0.5595	SVR uses regularization to prevent overfitting

7.4 Feature Importance Analysis

Table 4: Top 10 Most Influential Features (Linear Regression)

Feature	**Coefficient**
Loan Amount Request (USD)	34059.470448
Property Age	-24594.528834
Income (USD)	22755.768863
Credit Score	11622.244009
Income Stability	2518.287043
Profession_Pensioner	1770.431136
Property Price	-1273.515106
Type of Employment_Drivers	-1055.496666
Type of Employment_Laborers	-969.971154
Age	-613.671364

8 Best Practices

- **Data Quality:** Handled missing values appropriately using mean for numerical and mode for categorical variables
- **Feature Engineering:** Applied proper encoding techniques - label encoding for binary variables and one-hot encoding for multi-class categories
- **Scaling:** Used StandardScaler to normalize features, ensuring all variables contribute equally to the model
- **Data Splitting:** Implemented proper train/validation/test split (70/15/15) to avoid data leakage
- **Model Validation:** Used both holdout validation and K-fold cross-validation for robust performance assessment
- **Multiple Models:** Compared Linear Regression with SVR to identify the best performing approach
- **Comprehensive Evaluation:** Used multiple metrics (MAE, MSE, RMSE, R^2) for thorough performance assessment
- **Visualization:** Created comprehensive plots for data understanding and model interpretation
- **Hyperparameter Tuning:** Applied systematic hyperparameter optimization for SVR model
- **Documentation:** Maintained clear code structure with comments and proper variable naming

9 Learning Outcomes

- **End-to-end ML Pipeline:** Gained comprehensive understanding of the complete machine learning workflow from data loading to model evaluation
- **Data Preprocessing:** Learned effective techniques for handling missing values, encoding categorical variables, and feature scaling
- **Model Implementation:** Successfully implemented and compared multiple regression algorithms (Linear Regression and SVR)
- **Performance Evaluation:** Understood various evaluation metrics and their interpretations in the context of regression problems
- **Cross-Validation:** Learned the importance of K-fold cross-validation for robust model assessment
- **Hyperparameter Tuning:** Gained experience with systematic parameter optimization using RandomizedSearchCV
- **Data Visualization:** Developed skills in creating meaningful plots for exploratory data analysis and model interpretation
- **Model Interpretation:** Learned to interpret linear regression coefficients and understand feature importance
- **Overfitting Detection:** Understood how to identify and prevent overfitting through proper validation techniques
- **Scientific Reporting:** Enhanced skills in documenting and presenting machine learning experiments professionally

10 Conclusion

The Linear Regression model achieved an R^2 score of 0.5595 on the test set, explaining approximately 55.9% of the variance in loan sanction amounts. The most influential features were the loan amount request, property age, and income. The model shows good generalization with minimal overfitting (train R^2: 0.5783 vs test R^2: 0.5595). While SVR showed lower MAE, Linear Regression provided better overall performance with higher R^2 score and better interpretability. The comprehensive analysis demonstrates effective application of machine learning techniques for loan amount prediction.