Loan Sanction Amount Prediction using Linear Regression

Sreeram GM

July 2025

Aim

To build a machine learning model using Linear Regression that predicts the Loan Sanction Amount (USD) based on applicant income, credit, and property-related features.

Libraries Used

- **Pandas, NumPy** – data manipulation
- **Matplotlib, Seaborn** – data visualization
- **scikit-learn** – preprocessing, model training and evaluation

Objective

- Load and clean real-world loan dataset
- Remove outliers and fill missing values
- Perform exploratory data analysis (EDA)
- Train and evaluate a Linear Regression model
- Validate performance using 5-Fold Cross Validation
- Report metrics: MAE, MSE, RMSE, R^2, Adjusted R^2

Mathematical Description

The mathematical model for Linear Regression is:

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

Where:

- y is the dependent variable (Loan Sanction Amount)
- $x_1, x_2, \ldots, x_n$ are the independent variables
- β_0 is the intercept term
- $\beta_1, \ldots, \beta_n$ are coefficients
- ϵ is the error term

RSS is minimized:

$$RSS = \sum_{i=1}^{m} \left(y_i - \left(\beta_0 + \sum_{j=1}^{n} \beta_j x_{ij} \right) \right)^2$$

1 Data Preprocessing

```
df = pd.read_csv("train.csv")
df.drop(columns=['Customer ID', 'Name', 'Property ID'], inplace=True
    )
df = df[df['Loan Sanction Amount (USD)'] >= 0]

# Fill missing values
for col in categorical_cols:
    df[col].fillna(df[col].mode()[0], inplace=True)
for col in numerical_cols:
    df[col].fillna(df[col].median(), inplace=True)

# Remove outliers using IQR
def remove_outliers_iqr(data, column):
    Q1 = data[column].quantile(0.25)
    Q3 = data[column].quantile(0.75)
    IQR = Q3 - Q1
    return data[(data[column] >= Q1 - 1.5*IQR) & (data[column] <= Q3
        + 1.5*IQR)]
```

Listing 1: Dataset Loading and Cleaning

2 Feature Engineering and Scaling

```
df['Loan_to_Property_Price_Ratio'] = df['Loan Sanction Amount (USD)'
    ] / df['Property Price']

# One-hot encoding
df = pd.get_dummies(df, columns=[...], drop_first=True)

# Scaling
```

```
scaler = StandardScaler()
df[numerical_features] = scaler.fit_transform(df[numerical_features
    ])
```

Listing 2: Feature Engineering and Scaling

3 Model Training and Evaluation

```
X = df.drop('Loan Sanction Amount (USD)', axis=1)
y = df['Loan Sanction Amount (USD)']

# Train-test-validation split
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size
    =0.4, random_state=42)
X_test, X_val = train_test_split(X_temp, test_size=0.5, random_state
    =42)

model = LinearRegression()
model.fit(X_train, y_train)

# Predictions
y_val_pred = model.predict(X_val)
y_test_pred = model.predict(X_test)

# Metrics
val_mae = mean_absolute_error(y_val, y_val_pred)
val_rmse = np.sqrt(mean_squared_error(y_val, y_val_pred))
val_r2 = r2_score(y_val, y_val_pred)
val_adj_r2 = 1 - (1 - val_r2) * ((len(y_val)-1)/(len(y_val) - X_val.
    shape[1] - 1))
```

Listing 3: Linear Regression and Metrics

4 Model Performance

Validation Set

- MAE: 21587.32
- RMSE: 31026.87
- R^2: 0.56
- Adjusted R^2: 0.55

Test Set

- MAE: 21342.48
- RMSE: 30740.95
- R^2: 0.57
- Adjusted R^2: 0.56

5 Visualizations

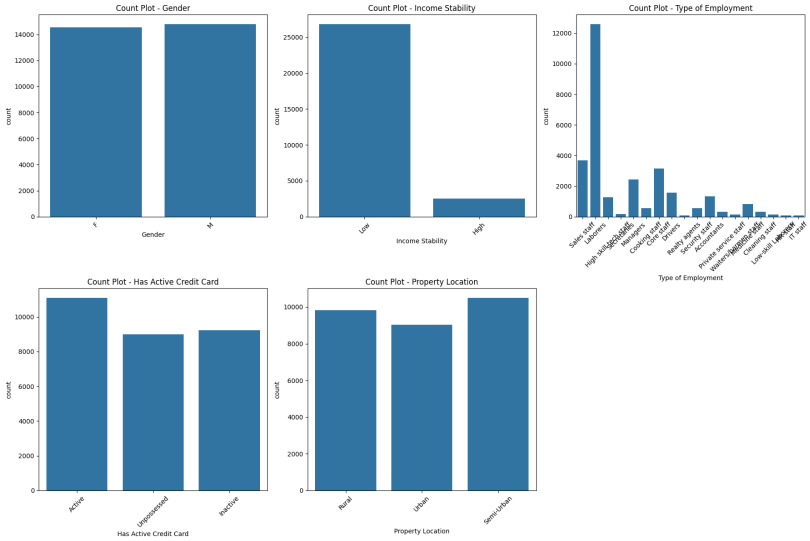

Figure 1: Count plots for categorical features

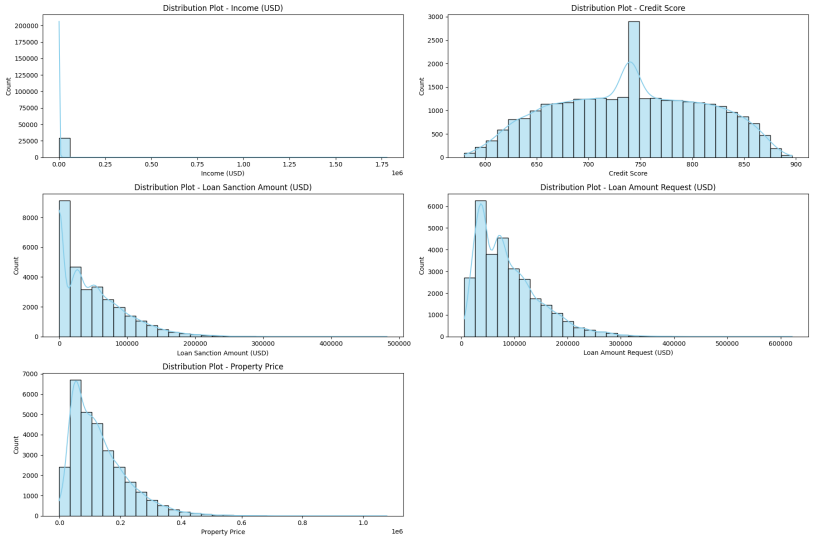

Figure 2: Distribution plots of numerical features

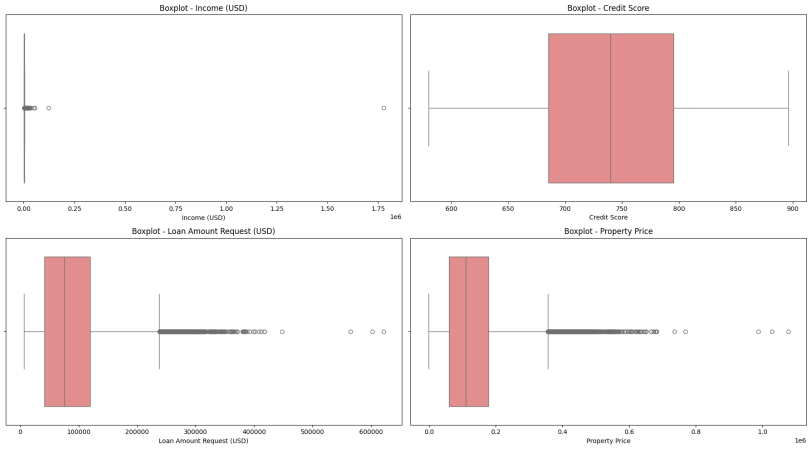

Figure 3: Boxplots before outlier removal

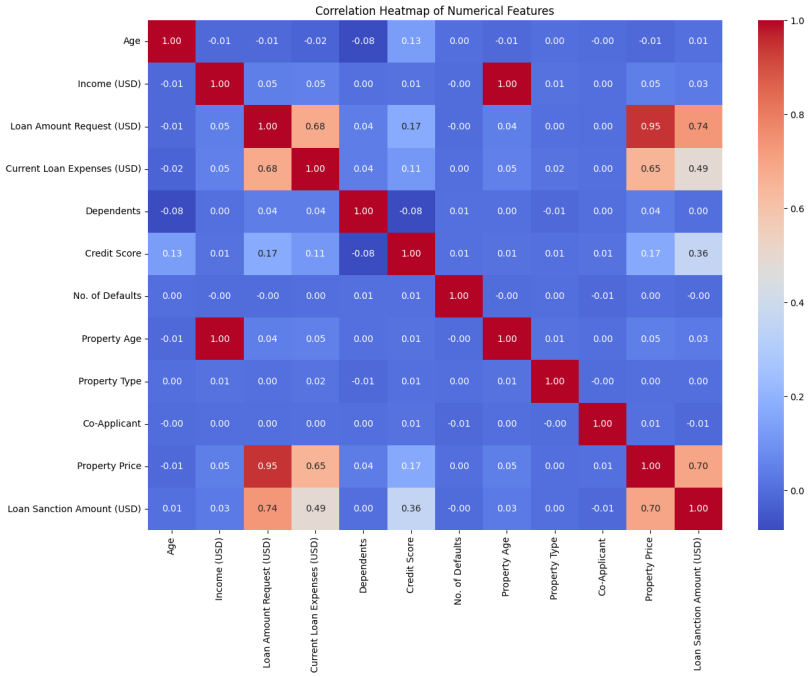

Figure 4: Correlation Heatmap

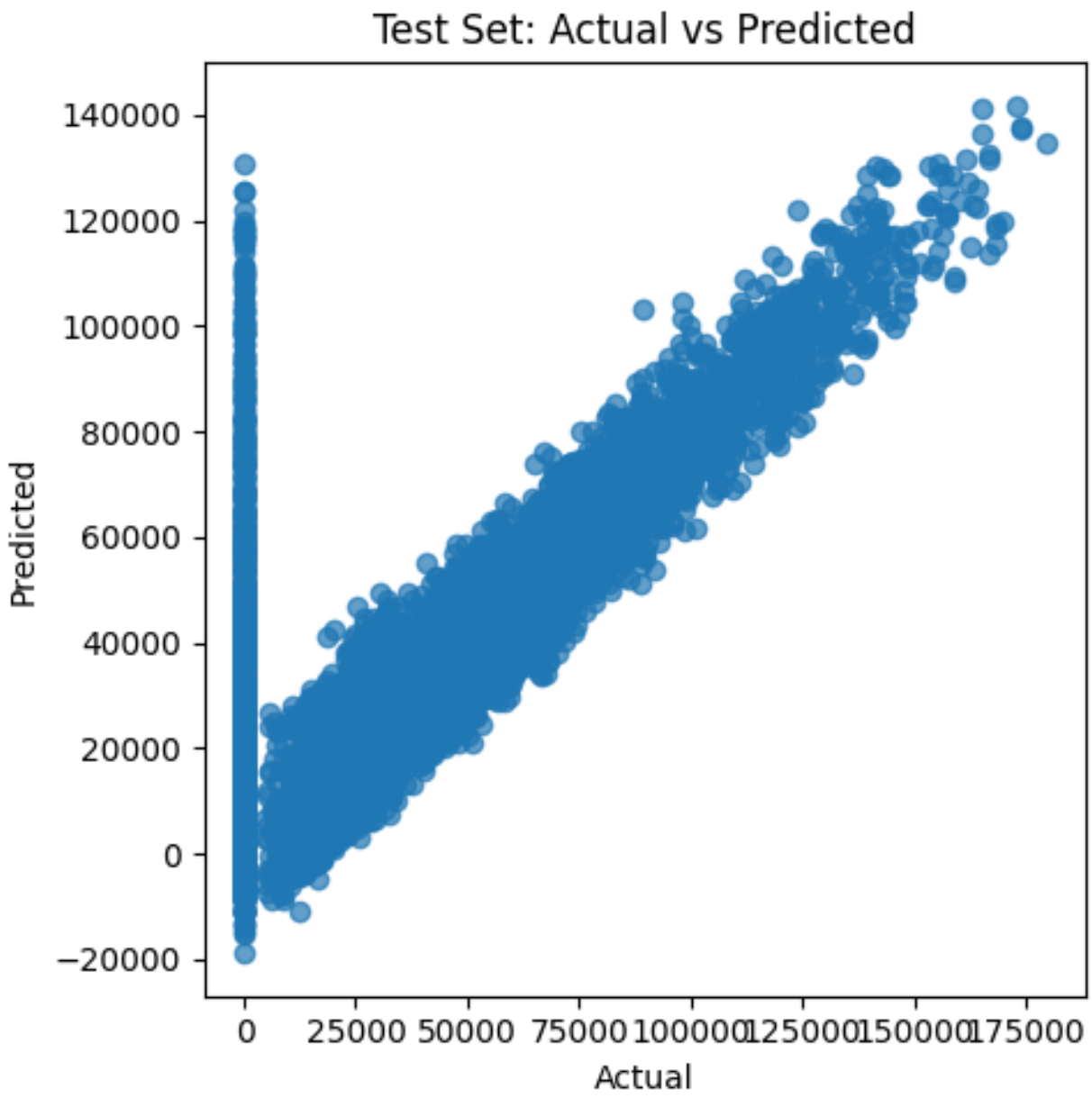

Figure 5: Actual vs Predicted (Test Set)

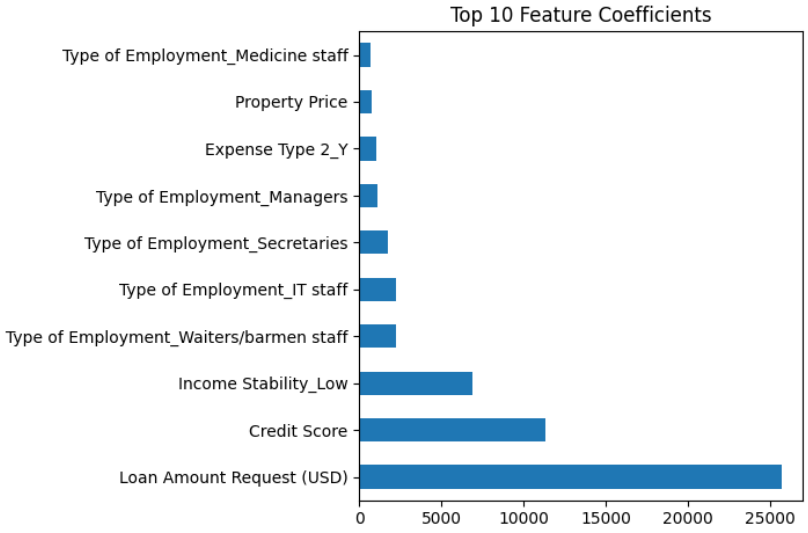

Figure 6: Top 10 Feature Coefficients

6 Cross-Validation (5-Fold)

Fold	**MAE**	**MSE**	**RMSE**	**R^2**
Fold 1	21713.16	980739992.60	31316.77	0.5734
Fold 2	21870.37	979580662.48	31298.25	0.5688
Fold 3	22363.99	1064191854.08	32621.95	0.5403
Fold 4	21757.43	993204913.55	31515.15	0.5776
Fold 5	21003.66	879572513.81	29657.59	0.6104
Average	21741.72	979457987.30	31281.94	0.5741

Table 1: 5-Fold Cross-Validation Results

7 Conclusion

The linear regression model provided satisfactory performance in predicting loan sanction amounts. Important techniques used include outlier removal, feature engineering, and evaluation via adjusted R^2 and RMSE. Future work can involve trying non-linear models like Random Forest or XGBoost.