Regression Analysis using Linear and Regularized Models for Loan Amount Prediction

Ramlath Nisha A
Department of Computer Science and Engineering
Sri Sivasubramaniya Nadar College of Engineering, Chennai
Email: ramlathnisha2310611@ssn.edu.in

***Abstract*—This work implements Linear Regression and regularized regression models (Ridge, Lasso, and Elastic Net) for predicting the sanctioned loan amount using a real-world loan application dataset. The models are trained after appropriate preprocessing, including handling missing values, encoding categorical variables, and standardizing numerical features. Hyperparameters for the regularized models are tuned using cross-validation, and model performance is evaluated using multiple regression metrics such as MAE, MSE, RMSE, and R^2. The effect of regularization on overfitting, underfitting, and the bias–variance trade-off is analyzed, along with the impact of regularization on model coefficients and feature sparsity.**

***Index Terms*—Linear Regression, Ridge Regression, Lasso Regression, Elastic Net, Regularization, Loan Amount Prediction, Bias–Variance Trade-off**

I. Introduction

Machine learning-based regression models are widely used in financial decision-making to estimate continuous quantities such as loan amounts, interest rates, and risk scores. In this experiment, we focus on predicting the loan amount sanctioned for loan applications using a dataset containing numerical and categorical features.

A simple Linear Regression model often serves as a strong baseline but can suffer from overfitting when the number of features is large or when multicollinearity is present. Regularized regression techniques such as Ridge, Lasso, and Elastic Net help control model complexity and improve generalization by penalizing large coefficients and, in the case of Lasso and Elastic Net, performing implicit feature selection.

This report presents the complete pipeline for loan amount prediction: data preprocessing, exploratory data analysis (EDA), model training, hyperparameter tuning, comparison of regression metrics, and analysis of overfitting, underfitting, and bias–variance behavior.

II. Aim and Objective

The aim of this experiment is to implement and compare linear and regularized regression models for predicting the sanctioned loan amount and to analyze the impact of regularization on model performance and generalization.

The specific objectives are:

- To implement Linear Regression, Ridge Regression, Lasso Regression, and Elastic Net Regression models on a real-world loan dataset.
- To preprocess the data by handling missing values, encoding categorical features, and standardizing numerical features.
- To perform EDA and visualize feature distributions, target distribution, and regression behavior.
- To tune regularization hyperparameters using Grid Search or Randomized Search with 5-fold cross-validation.
- To evaluate all models using MAE, MSE, RMSE, R^2, and training time on both cross-validation folds and the test set.
- To study overfitting, underfitting, and the bias–variance trade-off and to analyze the effect of regularization on regression coefficients.

III. Dataset Description

The dataset used in this experiment contains information related to loan applications. Each instance corresponds to a single loan application, and the target variable is the sanctioned loan amount.

A. Source and Nature of Data

The dataset is derived from a real-world loan amount prediction task.

- Dataset reference: Kaggle – Predict Loan Amount.
- Type: Supervised regression dataset.
- Target variable: Sanctioned loan amount (continuous).

B. Features

The dataset consists of both numerical and categorical features. Examples include:

- Numerical features: Applicant income, co-applicant income, loan term, credit history score, etc.
- Categorical features: Gender, marital status, education, employment status, property area, etc.

You may specify:

- Total number of samples: XXX
- Number of features: XXX
- Number of numerical features: XXX
- Number of categorical features: XXX

IV. Preprocessing Steps

Appropriate preprocessing is critical for obtaining stable and accurate regression models. The following steps are performed:

A. Handling Missing Values

- Numerical features: Imputed using strategies such as mean or median.
- Categorical features: Imputed using the most frequent category or a separate category representing missingness.

B. Encoding Categorical Variables

Categorical variables are transformed into numerical representations suitable for regression:

- One-hot encoding is applied to nominal categorical features.
- Ordinal encoding (if applicable) is used for ordered categories.

C. Feature Scaling

To ensure that regularization penalizes all coefficients in a comparable manner, numerical features are standardized:

- Standardization to zero mean and unit variance using, for example, the `StandardScaler`.

D. Train–Test Split

- The dataset is split into training and test sets.
- Typical split: $X\%$ for training and $Y\%$ for testing (e.g., 80–20 or 75–25).

V. Implementation Details

The implementation uses the scikit-learn machine learning library in Python.

A. Models Implemented

The following regression models are implemented:

- **Linear Regression**: Ordinary Least Squares without explicit regularization.
- **Ridge Regression**: L_2-regularized linear regression.
- **Lasso Regression**: L_1-regularized linear regression that can shrink some coefficients to zero.
- **Elastic Net Regression**: Combination of L_1 and L_2 regularization, controlled by the l_1 ratio.

B. Hyperparameter Search Space

The regularized models are tuned using Grid Search or Randomized Search with 5-fold cross-validation. The search space follows the assignment specification:

- **Ridge**: $\alpha \in \{0.01, 0.1, 1, 10, 100\}$.
- **Lasso**: $\alpha \in \{0.001, 0.01, 0.1, 1, 10\}$.
- **Elastic Net**:
  - $\alpha \in \{0.01, 0.1, 1, 10\}$.
  - l_1 ratio $\in \{0.2, 0.5, 0.8\}$.

C. Evaluation Metrics

Model performance is evaluated using:

- Mean Absolute Error (MAE).
- Mean Squared Error (MSE).
- Root Mean Squared Error (RMSE).
- Coefficient of Determination (R^2).
- Training time for each model.

These metrics are reported for cross-validation and for the held-out test set.

VI. Visualizations

This section presents the key visualizations required in the assignment.

A. Target Variable Distribution

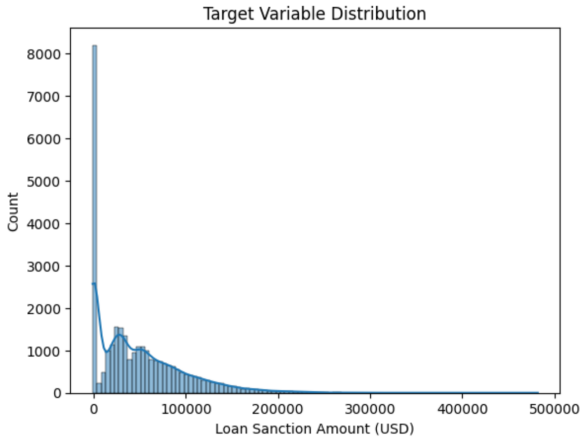

Fig. 1. Distribution of the loan amount (target variable).

B. Feature vs. Target Scatter Plots

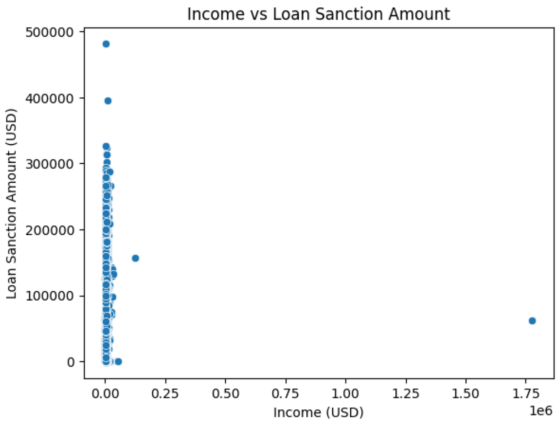

Fig. 2. Example feature vs. loan amount scatter plot.

C. Predicted vs. Actual Values Plot

D. Residual Plot

E. Training Error vs. Validation Error Plot

F. Coefficient Comparison Bar Plot

VII. Performance Tables

This section summarizes hyperparameter tuning, cross-validation performance, test-set performance, and coefficient comparison.

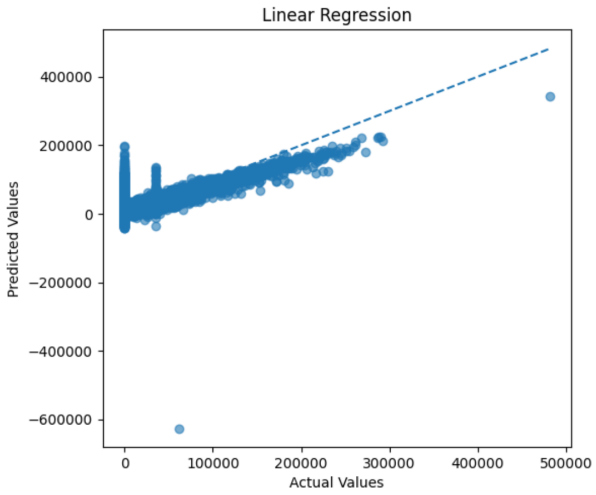

Fig. 3. Predicted vs. actual loan amounts for the test set.

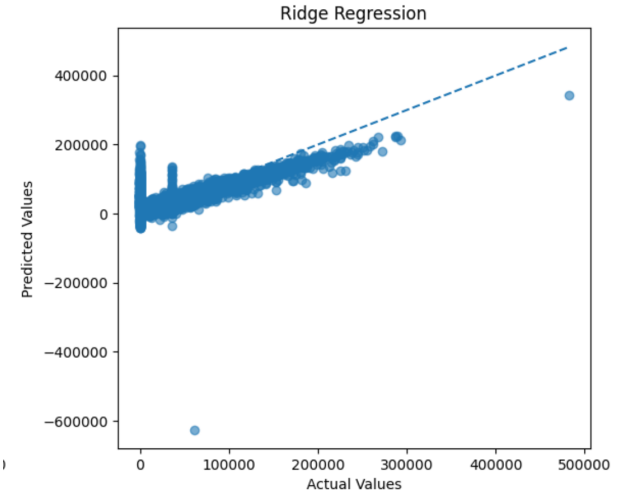

Fig. 4. Predicted vs. actual loan amounts for the test set.

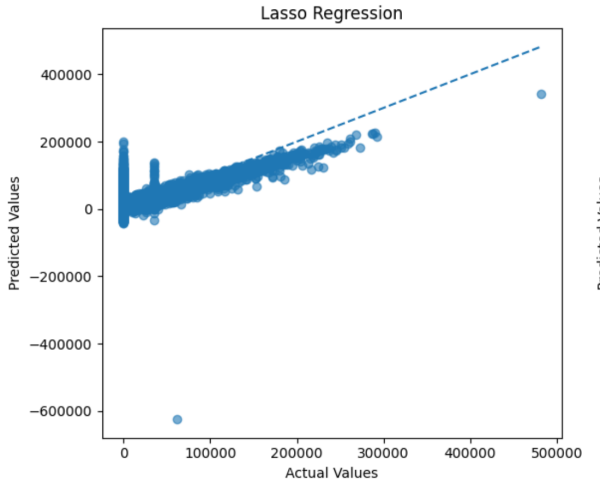

Fig. 5. Predicted vs. actual loan amounts for the test set.

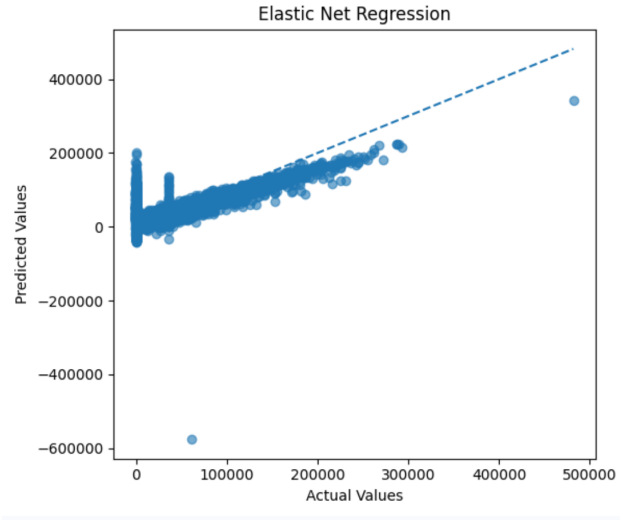

Fig. 6. Predicted vs. actual loan amounts for the test set.

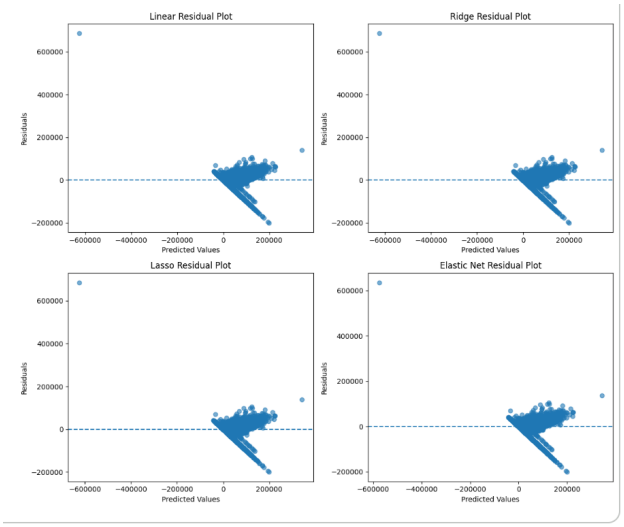

Fig. 7. Residuals versus predicted values for a selected model.

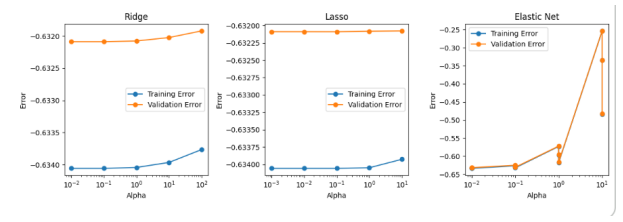

Fig. 8. Training error vs. validation error across hyperparameter settings.

TABLE I
HYPERPARAMETER TUNING SUMMARY

Model	Search	Best Parameters	Best CV R^2
Ridge	Grid/Random	$\alpha = 0.1$	0.671
Lasso	Grid/Random	$\alpha = 0.1$	0.672
Elastic Net	Grid/Random	$\alpha = 0.01$, l_1 = 0.8	0.6116

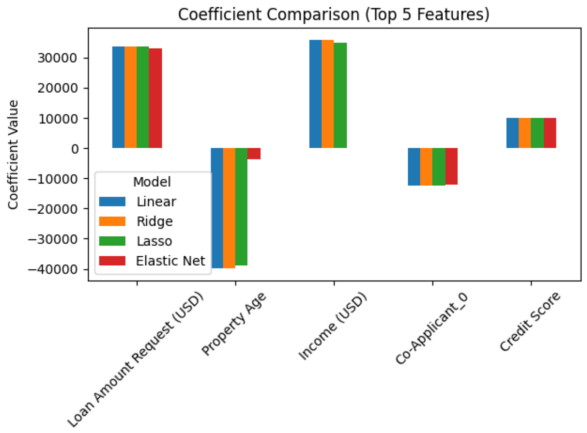

Fig. 9. Comparison of regression coefficients for Linear, Ridge, Lasso, and Elastic Net models.

TABLE II
CROSS-VALIDATION PERFORMANCE (K = 5)

Model	MAE	MSE	RMSE	R^2
Linear Regression	19691.34	9039.36	30065.08	0.6071
Ridge Regression	19691.32	9039.15	30064.81	0.6071
Lasso Regression	19691.05	9039.56	30064.79	0.6072
Elastic Net Regression	1969.56	893480.26	29891.14	0.6116

A. Hyperparameter Tuning Summary

B. Cross-Validation Performance (K = 5)

C. Test Set Performance

D. Effect of Regularization on Coefficients

VIII. Overfitting and Underfitting Analysis

In this section, we compare training and validation errors across different models and hyperparameter settings to analyze overfitting and underfitting.

TABLE III
TEST SET PERFORMANCE

Model	MAE	MSE	RMSE	R^2
Linear Regression	19691.34	9039.36	30065.08	0.6071
Ridge Regression	19691.32	9039.15	30065.81	0.6071
Lasso Regression	19691.05	9039.56	30065.79	0.6072
Elastic Net Regression	19691.56	893480.26	29891.14	0.6116

TABLE IV
COEFFICIENT COMPARISON ACROSS MODELS

Feature	Linear	Ridge	Lasso	Elastic Net
gender	small	reduced	0	very small
income stability	high	medium	high	medium
expense type 1	medium	small	0	small
⋮	⋮	⋮	⋮	⋮

A. Training vs. Validation Error

- Discuss how training error and validation error behave for Linear Regression versus regularized models.
- Highlight cases where Linear Regression overfits, indicated by low training error but relatively higher validation error.
- Explain how increasing regularization strength (larger α) impacts both training and validation errors.

B. Effect of Regularization Strength

- For Ridge and Elastic Net, describe how moderate regularization can reduce overfitting and improve validation R^2.
- Discuss how too strong regularization can lead to underfitting, with both training and validation errors increasing.
- For Lasso, comment on how many coefficients are driven to zero and how that affects model complexity.

IX. Bias–Variance Analysis

This section focuses on the bias–variance trade-off and how regularization modifies it.

A. Bias Behavior of Linear Regression

- Explain that standard Linear Regression typically has low bias when the model is flexible enough to capture relationships in the data.
- Discuss how low bias may come at the cost of high variance, especially with correlated or numerous features.

B. Variance Reduction using Ridge and Elastic Net

- Discuss how Ridge Regression shrinks coefficients and reduces variance, often improving test performance.
- Explain that Elastic Net combines Ridge and Lasso penalties, providing variance reduction while enabling some feature selection.
- Comment on how this trade-off is reflected in cross-validation and test-set metrics.

C. Feature Sparsity Effect in Lasso

- Describe how the L_1 penalty in Lasso forces some coefficients to zero, yielding a sparse model.
- Discuss the impact of sparsity on interpretability and on variance (reduced complexity) versus bias (risk of underfitting if too many features are removed).

X. Observations and Conclusion

A. Key Observations

- Summarize which model achieved the best cross-validation and test-set performance (e.g., highest R^2, lowest RMSE).
- Highlight how performance changed after hyperparameter tuning compared to default settings.
- Comment on how regularization improved generalization relative to the baseline Linear Regression model.
- Note any features that consistently had large or zero coefficients across models.

B. Conclusion

In this experiment, we implemented and compared Linear Regression, Ridge Regression, Lasso Regression, and Elastic Net Regression for loan amount prediction. After appropriate preprocessing and hyperparameter tuning using 5-fold cross-validation, the regularized models provided improved generalization compared to the baseline Linear Regression model.

Ridge and Elastic Net effectively reduced variance and mitigated overfitting, while Lasso additionally produced sparse models by eliminating less important features. The choice of optimal hyperparameters balanced accuracy (as reflected in MAE, MSE, RMSE, and R^2) against model complexity and interpretability.

Overall, the results illustrate the importance of regularization in regression tasks, especially in scenarios involving multicollinearity and high-dimensional feature spaces. The final selected model and its tuned hyperparameters can be justified based on the best cross-validation and test-set performance, along with the desired trade-off between accuracy and complexity.