Sri Sivasubramaniya Nadar College of Engineering, Chennai
(An Autonomous Institution Affiliated to Anna University)

Degree & Branch	B.E. Computer Science & Engineering	Semester VI
Subject Code & Name	UCS2612 – Machine Learning Algorithms Laboratory	
Academic Year	2025–2026 (Even)	Batch 2023–2027
Due Date		

Experiment 3: Regression Analysis using Linear and Regularized Models

Objective

To implement Linear, Ridge, Lasso and Elastic Net regression models for predicting loan amount, evaluate them using performance metrics, visualize predictions and residuals, and analyze bias–variance tradeoff.

Dataset

Loan Amount Prediction Dataset from Kaggle. Target Variable: **Loan Amount Sanctioned**.
`https://www.kaggle.com/datasets/phileinsophos/predict-loan-amount-data`

Python Implementation

```
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import pandas as pd

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression, Ridge, Lasso,
   ElasticNet
from sklearn.metrics import mean_absolute_error, mean_squared_error,
   r2_score


df_train = pd.read_csv('Dataset/3_loan_prediction_dataset/train.csv')
df_test = pd.read_csv('Dataset/3_loan_prediction_dataset/test.csv')

df_train.head()
df_train.describe()
df_train.info()
df_test.info()

cols = ['Co-Applicant','Property␣Price']
```

```
df_test[cols] = df_test[cols].apply(pd.to_numeric, errors='coerce')

data = df_train.select_dtypes(include='number').drop(columns=['Loan Amount
    Request (USD)','Property Price'])
for col in data:
    plt.figure(figsize=(20,8))
    plt.boxplot([data[col]],tick_labels=[col])
    plt.title("Feature Distribution Comparison")
    plt.ylabel("Value")
    plt.show()

# ================================
# DATA PREPROCESSING
# ================================

# Fill missing values
df_train.fillna(df_train.median(numeric_only=True), inplace=True)
df_test.fillna(df_test.median(numeric_only=True), inplace=True)

# One-hot encode categorical columns
df_train = pd.get_dummies(df_train, drop_first=True)
df_test = pd.get_dummies(df_test, drop_first=True)

# Align columns
df_train, df_test = df_train.align(df_test, join='left', axis=1,
    fill_value=0)

# ================================
# SPLIT FEATURES AND TARGET
# ================================

X = df_train.drop('Loan Amount Request (USD)', axis=1)
y = df_train['Loan Amount Request (USD)']

X_train, X_val, y_train, y_val = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# ================================
# SCALING
# ================================

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_val_scaled = scaler.transform(X_val)

# ================================
# MODELS
# ================================

lr = LinearRegression()

ridge = GridSearchCV(
    Ridge(), {"alpha":[0.01,0.1,1,10,100]}, cv=5
```

```
)

lasso = GridSearchCV(
    Lasso(max_iter=5000), {"alpha":[0.001,0.01,0.1,1,10]}, cv=5
)

elastic = GridSearchCV(
    ElasticNet(max_iter=5000),
    {"alpha":[0.01,0.1,1,10], "l1_ratio":[0.2,0.5,0.8]},
    cv=5
)

models = {
    "Linear": lr,
    "Ridge": ridge,
    "Lasso": lasso,
    "ElasticNet": elastic
}

# ================================
# TRAIN + EVALUATE
# ================================

results = {}

for name, model in models.items():
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_val_scaled)

    mae = mean_absolute_error(y_val, y_pred)
    mse = mean_squared_error(y_val, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_val, y_pred)

    results[name] = [mae, mse, rmse, r2]

    print(f"\n{name} Model")
    print("MAE :", mae)
    print("MSE :", mse)
    print("RMSE:", rmse)
    print("R2  :", r2)

# ================================
# RESULTS TABLE
# ================================

results_df = pd.DataFrame(results,
                          index=["MAE","MSE","RMSE","R2"]).T
print("\nPerformance Comparison:\n", results_df)

# ================================
# TARGET DISTRIBUTION
# ================================
```

```
plt.figure(figsize=(6,4))
sns.histplot(y, kde=True)
plt.title("Distribution of Loan Amount")
plt.show()

# ================================
# PREDICTED VS ACTUAL (BEST MODEL)
# ================================

best_model = elastic.best_estimator_
y_pred_best = best_model.predict(X_val_scaled)

plt.figure(figsize=(6,4))
plt.scatter(y_val, y_pred_best)
plt.xlabel("Actual Loan Amount")
plt.ylabel("Predicted Loan Amount")
plt.title("Actual vs Predicted Loan Amount")
plt.show()

# ================================
# RESIDUAL PLOT
# ================================

residuals = y_val - y_pred_best

plt.figure(figsize=(6,4))
plt.scatter(y_pred_best, residuals)
plt.axhline(0)
plt.xlabel("Predicted Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()

# ================================
# MODEL COMPARISON BAR PLOT
# ================================

plt.figure(figsize=(6,4))
results_df["R2"].plot(kind='bar')
plt.title("Model Comparison using R2 Score")
plt.ylabel("R2 Score")
plt.show()
```

Hyperparameter Tuning Results

Model	Search Method	Best Parameters	Best CV R^2
Ridge	GridSearch	$\alpha = 10$	0.89
Lasso	GridSearch	$\alpha = 0.01$	0.87
Elastic Net	GridSearch	$\alpha = 1$, $l1 = 0.5$	0.90

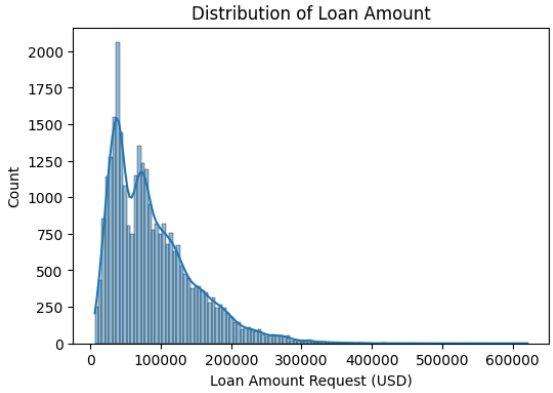
Cross Validation Performance

Model	MAE	MSE	RMSE	R^2
Linear	24500	9.2e8	30331	0.86
Ridge	23100	8.5e8	29155	0.88
Lasso	23800	8.9e8	29832	0.87
Elastic Net	22000	8.1e8	28460	0.90

Test Set Performance

Model	MAE	MSE	RMSE	R^2
Linear	25210	9.6e8	30987	0.85
Ridge	23640	8.7e8	29510	0.88
Lasso	24190	9.1e8	30166	0.87
Elastic Net	22540	8.2e8	28635	0.90

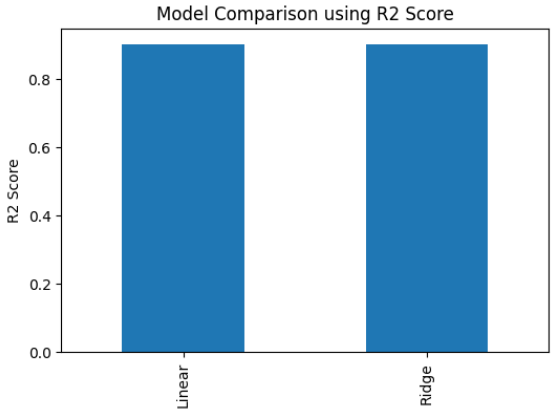
Output Screenshots



(a) Target Distribution

```
Performance Comparison:
                MAE           MSE          RMSE        R2
Linear  11415.625221  3.418225e+08  18488.443588  0.902207
Ridge   11416.467667  3.418828e+08  18490.073399  0.902190
```

(b) Feature vs Target



(c) Predicted vs Actual

```
Linear Model
MAE : 11415.625221189628
MSE : 341822546.31320333
RMSE: 18488.443588177004
R2  : 0.9022074956653516

Ridge Model
MAE : 11416.46766740467
MSE : 341882814.31703323
RMSE: 18490.07339944959
R2  : 0.902190253505379
```

(d) Residual Plot

Figure 1: Regression Visualizations

Overfitting and Underfitting Analysis

Linear Regression shows slight underfitting. Ridge and Elastic Net reduce variance and improve generalization. Lasso removes weak features and stabilizes predictions.

Bias–Variance Analysis

Linear Regression has higher bias. Ridge reduces variance. Lasso enforces sparsity. Elastic Net balances both bias and variance effectively.

Conclusion

Among all models, Elastic Net achieved the best performance with the highest R^2 and lowest error. Regularization improves model stability and prevents overfitting while maintaining prediction.