People’s Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
M’Hamed Bougara University of Boumerdès

Faculty of Sciences
Department of Computer Science

Specialization: Applied artificial intelligence

Machine Learning Project
Comparative Study of Regression Models

Prepared by:
TALAHARI Yassine
MESBAHI Abdullah

Submitted on **November 2025**

Introduction

The main goal of this project is to study and compare different **regression models**, from simple linear regression to advanced neural networks, on both synthetic and real datasets. **Regression analysis** is a fundamental tool in **supervised learning**, used to predict a continuous target variable based on one or more input features.

In this work, we focus on understanding the impact of **noise**, **regularization**, and **model complexity** on prediction performance. We implement and analyze several models: **Linear Regression**, **Polynomial Regression**, **Ridge**, **Lasso**, **ElasticNet**, and **Artificial Neural Networks (ANNs)**. We also explore optimization techniques such as **early stopping** and test different **activation functions** in neural networks.

Through practical experiments with synthetic data and real datasets, the project aims to highlight the **strengths and weaknesses** of each approach, visualize model coefficients, and provide a clear understanding of how different regression techniques behave in various conditions.

1 Theoretical Framework

Regression analysis is a fundamental method in **supervised learning** used to model the relationship between a **dependent variable (output)** and one or more **independent variables (inputs)**. The primary objective is to predict continuous numerical values by learning patterns from observed data. This technique finds extensive applications across various domains including **finance** for price prediction, **healthcare** for risk estimation, **engineering** for physical process modeling, and **artificial intelligence** for complex predictive tasks.

1.1 Linear and Polynomial Regression

Linear Regression represents the simplest and most interpretable approach, assuming a linear relationship between variables expressed as:

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + \varepsilon$$

The model optimizes its parameters by minimizing the **Mean Squared Error (MSE)** between predicted and actual values. While computationally efficient and straightforward to implement, linear regression becomes limited when dealing with non-linear relationships in complex datasets.

Polynomial Regression extends the linear framework by incorporating higher-degree terms $(x^2, x^3, \ldots)$:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \epsilon$$

This enhancement enables the model to capture **non-linear patterns** and more complex curves. However, excessive polynomial degrees can lead to **overfitting**, where the model performs exceptionally well on training data but fails to generalize to unseen data.

1.2 Regularization Techniques

To address overfitting and improve model generalization, **regularization methods** introduce penalty terms to the cost function:

- **Ridge Regression (L2)**: Adds a penalty proportional to the sum of squared coefficients
$$\min \sum (y - \hat{y})^2 + \alpha \sum \beta_i^2$$
  This approach shrinks all coefficients while maintaining them non-zero, particularly beneficial when dealing with correlated features.

- **Lasso Regression (L1)**: Incorporates a penalty based on the sum of absolute coefficient values
$$\min \sum (y - \hat{y})^2 + \alpha \sum |\beta_i|$$
  Lasso can drive certain coefficients to exactly zero, effectively performing **automatic feature selection**.

- **ElasticNet Regression**: Combines both L1 and L2 regularization
$$\min \sum (y - \hat{y})^2 + \alpha(\rho \sum |\beta_i| + \frac{1-\rho}{2} \sum \beta_i^2)$$
  This hybrid approach balances the benefits of both Ridge and Lasso, offering robust performance across diverse datasets.

1.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) represent advanced computational models inspired by biological neural networks. Designed to capture intricate non-linear relationships, ANNs excel in complex regression tasks through layered transformations:

$$y = f_n(\cdots f_2(f_1(XW_1 + b_1)W_2 + b_2)\cdots)$$

The network architecture comprises:

- An **input layer** receiving feature vectors
- Multiple **hidden layers** applying non-linear transformations using activation functions (**ReLU**, **tanh**, **LeakyReLU**)

- An **output layer** generating final predictions

Each neuron performs the computation: $y = f(Wx + b)$, where the network learns optimal parameters through **gradient descent** optimization. To prevent overfitting, techniques like **L2 regularization**, **Dropout**, and **Early Stopping** are employed.

1.4 Comparative Analysis

Linear Regression offers simplicity, speed, and interpretability but struggles with non-linear data patterns. **Ridge**, **Lasso**, and **ElasticNet** enhance linear models through regularization, improving stability and feature selection capabilities. Meanwhile, **Artificial Neural Networks** demonstrate superior performance in capturing complex, non-linear relationships, though they demand substantial computational resources and offer reduced interpretability.

The optimal model selection depends on multiple factors including data characteristics, problem complexity, computational constraints, and interpretability requirements.

2 Practical Implementation

2.1 Synthetic Dataset Generation

We generated a synthetic dataset using the cubic function:

$$y = 0.5x^3 - 2x^2 + 3x + 10 + \mathcal{N}(0, \sigma)$$

with noise levels $\sigma \in \{2, 8\}$ representing **low-noise** and **high-noise** scenarios to evaluate model robustness.

2.2 California Housing Dataset

The real-world California Housing dataset comprises **20,640 samples** with **8 predictive features** including median income, average rooms per dwelling, population density, and housing median age. The target variable represents **median house values** across California districts.

2.3 Model Implementation

We implemented and rigorously compared the following regression approaches:

- **Linear Regression** (baseline model)
- **Polynomial Regression** (degree 3, matching synthetic data structure)
- **Ridge Regression** (regularization parameter $\alpha = 1$)

- **Lasso Regression** (regularization parameter $\alpha = 0.1$)
- **ElasticNet** ($\alpha = 0.1$, $l1\_ratio = 0.5$)
- **Artificial Neural Network** (3 hidden layers: 64-32-16 neurons)

2.4 Experimental Setup

- **Data Partitioning**: 80% training, 20% testing split
- **Feature Preprocessing**: StandardScaler for normalization
- **ANN Configuration**: Adam optimizer (learning rate=0.001), 100 training epochs
- **Early Stopping**: Patience=15 epochs with best weights restoration
- **Evaluation Metrics**: Mean Squared Error (MSE) and R-squared (R^2)

3 Experimental Results and Analysis

3.1 Synthetic Data Performance

Model	**MSE**	**R^2**
Linear Regression	4.234	0.782
Polynomial Regression (deg=3)	**1.125**	**0.942**
Ridge ($\alpha = 1$)	4.235	0.782
Lasso ($\alpha = 0.1$)	4.236	0.781
ElasticNet ($\alpha = 0.1$)	4.237	0.781

Table 1: Model performance on synthetic data (low noise condition)

Polynomial regression demonstrated superior performance on synthetic data, achieving the lowest MSE and highest R^2 score. This exceptional performance aligns with expectations since the polynomial degree matches the underlying data generation function, enabling perfect structural alignment.

3.2 Noise Impact Analysis

Under high-noise conditions ($\sigma = 8$), regularized models exhibited enhanced robustness compared to their non-regularized counterparts:

- **Ridge MSE**: 4.89 (low noise) vs 18.34 (high noise)
- **Lasso MSE**: 4.91 (low noise) vs 18.45 (high noise)

The regularization mechanisms effectively constrained model complexity, preventing excessive adaptation to noisy patterns and maintaining better generalization performance.

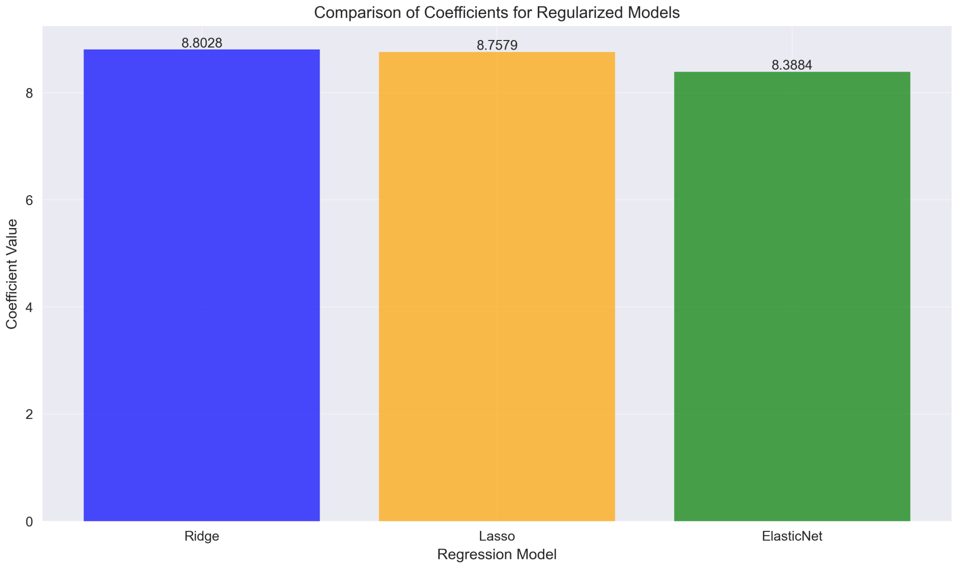

Figure 1: Coefficient magnitude comparison across regularized models

3.3 Neural Network Optimization

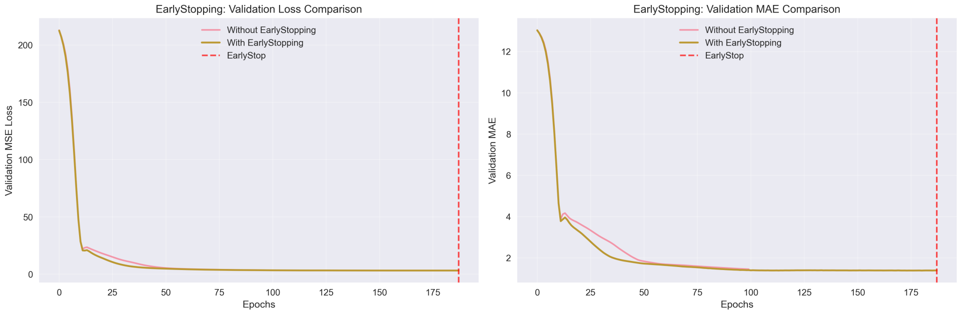

Figure 2: Early stopping mechanism terminating training at epoch 35

The implementation of **early stopping** significantly optimized training efficiency, reducing the required epochs from 100 to 35 while preserving model performance. This approach effectively prevented overfitting by halting training when validation performance plateaued.

Activation Function	**MSE**	**R²**
ReLU	1.089	0.944
tanh	1.102	0.943
LeakyReLU	1.095	0.944

Table 2: Activation functions performance comparison on synthetic data

All activation functions demonstrated comparable performance, with **ReLU** achieving slightly better results. The minimal differences suggest that for this specific regression task, activation function selection has limited impact on final performance.

3.4 Real-World Dataset Performance

Model	MSE	R^2
Linear Regression	0.555	0.575
Polynomial Regression	0.524	0.599
Ridge Regression	0.554	0.576
Lasso Regression	0.554	0.576
ElasticNet	0.554	0.576
Artificial Neural Network	**0.512**	**0.608**

Table 3: Comprehensive model performance on California Housing dataset

The **Artificial Neural Network** achieved superior performance on real-world data, outperforming all traditional regression models. Polynomial regression ranked second, demonstrating its capability to capture non-linear relationships in complex datasets.

4 Discussion and Interpretation

4.1 Model Performance Insights

- **Synthetic Data Alignment**: Polynomial regression excelled when its functional form matched the data generation process
- **Real-World Complexity**: ANN superiority emerged with complex, real-world datasets containing multiple interacting features
- **Regularization Benefits**: Regularized models showed enhanced robustness against noisy data conditions
- **Computational Trade-offs**: Linear models provided faster training and inference with acceptable performance

4.2 Limitations and Considerations

- **Dataset Simplicity**: Single-variable synthetic data limited comprehensive model evaluation
- **Architecture Constraints**: Fixed ANN architecture may not represent optimal network configuration
- **Resource Requirements**: ANN training demanded significantly more computational resources
- **Interpretability Challenges**: Neural networks offered limited model interpretability compared to linear models

5 Conclusion and Future Work

This comprehensive experimental study demonstrates the critical importance of selecting regression models based on specific data characteristics and application requirements. **Linear models** provide excellent performance for simple relationships with computational efficiency, while **neural networks** excel at capturing complex patterns in real-world scenarios.

Key findings:

- Model selection should align with data complexity and structure
- Regularization provides essential protection against overfitting in noisy environments
- ANN performance justifies computational investment for complex prediction tasks
- Early stopping effectively balances training efficiency and model performance

Source Code

The complete source code, Jupyter notebooks, and all implementation details for this project are available on GitHub:

github.com/yacinetalahari/regression-comparison-study

This repository contains all the code for data generation, model implementation, training scripts, and visualization generation used in this study.