Lasso Regression with Neural Network-Based Nonlinear Corrections

Preparation for Master Thesis

Erdem Koca

March 2025

1 Introduction

This report lays the theoretical groundwork for the master thesis and helps define its research direction.

The literature review focused on Chapter 3 of *The Elements of Statistical Learning* [1], covering Linear Regression, along with related topics like Random Forest (Chapter 15) and Neural Networks (Chapter 11).

Additionally, practical work was done using PyTorch for neural network implementation. The potential of combining linear regression with neural networks for non-linear corrections remains an open question for further study.

2 Literature Review

2.1 Linear Regression Idea

Linear regression aims to minimize prediction error, commonly achieved using the Least Squares method. It is a type of supervised learning where the model learns from labeled data to predict an output variable.

Linear regression assumes that the relationship between the input variables $X_1, X_2, \ldots, X_p$ and the output variable Y is linear:

$$E(Y|X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p. \tag{1}$$

Even if the true relationship is non-linear, linear approximations can still be useful due to their simplicity and effectiveness.

Despite advancements in computing that enable complex non-linear models (e.g., deep learning, gradient boosting), linear models remain widely used due to:

- Simplicity and Interpretability.
- Strong Performance on Small Datasets.

2.2 Subset Selection

Subset selection improves **prediction accuracy** and **interpretability** by selecting only the most relevant variables in regression models. Least squares regression often suffers from **high variance**, and removing unnecessary features reduces this variance, enhancing generalization.

2.2.1 Methods of Subset Selection

Best-Subset Selection: Finds the subset minimizing **RSS**, but is computationally expensive.

Stepwise Selection: Iteratively adds (**Forward**) or removes (**Backward**) features. Faster but not always optimal.

Forward-Stagewise Regression: Gradually adjusts coefficients, useful in high-dimensional settings but slow.

2.3 Shrinkage Methods

Shrinkage methods reduce variance and improve generalization by shrinking regression coefficients, preventing overfitting. They offer an alternative to **subset selection**, which fully includes or excludes variables.

2.3.1 Ridge Regression

Ridge regression extends Least Squares by adding an **L2 penalty** to shrink coefficients, reducing variance without setting them to zero:

$$\hat{\beta}^{ridge} = \arg\min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2. \tag{2}$$

It has a **closed-form solution**:

$$\hat{\beta}^{ridge} = (X^T X + \lambda I)^{-1} X^T y. \tag{3}$$

The regularization parameter λ controls shrinkage, balancing bias and variance.

2.3.2 Lasso

Lasso regression performs **feature selection** by applying an **L1 penalty**, shrinking some coefficients to exactly zero:

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq t. \tag{4}$$

Unlike Ridge, which shrinks all coefficients, Lasso forces some to zero due to the **sharp, non-differentiable** L_1 penalty. The regularization parameter λ controls sparsity, with higher values eliminating more features. The optimal λ is chosen via **cross-validation**.

2.3.3 Comparison Table

Feature	**Least Squares**	**Ridge Regression (L2)**	**Lasso Regression (L1)**
Feature Selection	✗ No	✗ No	✓ Yes ($\beta_j = 0$)
Shrinkage Type	✗ None	✓ Shrinks coefficients	✓ Shrinks & sets some to zero
Mathematical Solution	✓ Closed-form	✓ Closed-form	✗ Iterative solution
Effect on Model	Overfits with many features	Reduces variance but keeps all features	Reduces variance & selects important features

Table 1: Comparison of LS, Ridge, and Lasso regression

2.3.4 Group Lasso

Group Lasso extends Lasso by introducing **structured sparsity**, applying an **L2 penalty within groups** of variables while maintaining an **L1 penalty across groups**. This enforces **group-wise sparsity**, meaning entire groups of coefficients can be set to zero when λ is large.

The optimization problem is:

$$\hat{\beta}^{group} = \arg\min_{\beta \in \mathbb{R}^p} \left(\left\| y - \beta_0 \mathbf{1} - \sum_{\ell=1}^{L} \mathbf{X}_\ell \beta_\ell \right\|_2^2 + \lambda \sum_{\ell=1}^{L} \sqrt{p_\ell} \|\beta_\ell\|_2 \right). \tag{5}$$

Here, $\mathbf{X}_\ell$ represents predictors in the ℓ-th group, β_ℓ is the corresponding coefficient vector, and $\|\beta_\ell\|_2$ is its L2 norm.

Key Insights:

- If a group is removed, all its predictors are excluded.
- Unlike Lasso, which selects individual features, Group Lasso operates on entire groups.
- Useful for **biological pathways**, **categorical predictors**, and **time-series segments**.

Comparison with Lasso:

Method	Regularization	Effect
Lasso	$\sum \lvert\beta_j\rvert$ (L1 norm)	Selects individual features
Group Lasso	$\sum \lVert\beta_\ell\rVert_2$ (L1+L2 norm)	Selects entire groups of variables

Table 2: Comparison of Lasso and Group Lasso

2.3.5 Further Properties of Lasso

Lasso is widely studied for its **feature selection** and **model recovery** capabilities.

Lasso performs well in **high-dimensional settings** where $p > N$ and improves as N and p grow. Under specific conditions (e.g., **irrepresentable condition**), it can consistently select the correct predictors.

Lasso **biases nonzero coefficients toward zero** due to shrinkage, unlike Least Squares, which is unbiased but has high variance. To reduce this bias, **Relaxed Lasso** (Meinshausen, 2007) is used:

1. Runs Lasso to select features.
2. Refits an Least Squares model using only the selected features.

2.3.6 Random Forest Feature Importance

Random Forest (RF) is an learning method that improves predictive accuracy by averaging multiple decision trees trained on bootstrapped data with **random feature selection**. This reduces overfitting and captures **nonlinear relationships**.

Comparison with Lasso:

RF assigns **feature importance scores** based on tree splits. However, RF tends to distribute importance across correlated variables rather than removing them entirely.

Lasso is preferable for **feature selection** and high-dimensional data, while RF outperform in capturing **complex patterns** but is computationally more expensive. Though both reduce variance, Lasso does so by penalizing coefficients, whereas RF averages multiple models.

- **Lasso:** Selects key features by setting some coefficients to zero.
- **Random Forest:** Spreads importance across correlated features.
- **Lasso vs. RF:** Lasso is more interpretable and suited for high-dimensional data, while RF captures nonlinear interactions.

3 Proposed Workplan for the Thesis

3.1 Concept: Fusion of Lasso and Neural Networks

This thesis investigates the integration of **Lasso regression** with **neural networks** to improve predictive performance by leveraging Lasso for feature selection and neural networks for non-linear corrections.

Key Idea:

- Lasso is used to **select relevant features** by shrinking some coefficients to exactly zero.
- A neural network learns a **non-linear correction function** that adjusts the remaining Lasso-selected coefficients.
- The correction function is defined as:

$$\beta_j = g(x_{-j}) = 1 + \tilde{g}(x_{-j}) \tag{6}$$

  where:

  - x_{-j} represents **all selected features except** x_j.
  - $\tilde{g}$ is a neural network that models interactions between selected features.
  - $\tilde{g}(0) = 0$ ensures that the correction is neutral for an average input.
  - $\tilde{g}(x_{-j}) \in (-1, 1)$ limits correction magnitude, maintaining interpretability.

3.2 Methodology

The study will:

- Implement **Lasso regression** for sparse feature selection.
- Develop a **neural network** that modifies Lasso-selected coefficients based on interactions with other selected variables.
- Ensure interpretability by applying constraints: $\tilde{g}(0) = 0$ and $\tilde{g}(x_{-j}) \in (-1, 1)$.
- Compare with alternative methods such as **Random Forest** to evaluate performance trade-offs.
- Use **PyTorch** for model development and training.

3.3 Challenges

- **Integrating Lasso and NN effectively** while maintaining model interpretability.
- Ensuring that the **NN corrections remain controlled** and do not disrupt sparsity.
- **Stabilizing the training process** to avoid oscillations between Lasso and NN optimization.

3.4 Evaluation Metrics

The model will be assessed using:

- **Mean Squared Error (MSE)** to measure regression accuracy.
- **Feature selection stability** to ensure Lasso selects consistent variables.
- **Interpretability**: ensuring that Lasso coefficients remain meaningful even after NN-based corrections.

3.5 Key Questions to Address

- What is the best structure for the neural network to provide effective corrections while preserving sparsity?
- What training procedure ensures convergence and stability?
- How does this approach compare to classical alternatives like Random Forest in terms of performance and interpretability?

References

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2009.