Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Comparative Analysis of Regularized Linear Models and Deep Neural Networks for High-Dimensional Predictive Regression

Deepthi Varikuti

varikutideepthi@gmail.com

Koneru Lakshmaiah Education Foundation

Raghavendra Sai Nallagatla

Koneru Lakshmaiah Education Foundation

Article

Keywords: High-Dimensional Data, Regularized Regression, Deep Neural Networks, Predictive Modelling, Model Comparison, Ridge Regression, Lasso, Elastic Net, Machine Learning

Posted Date: November 24th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11410745/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

High-dimensional predictive regression has become increasingly important in modern data-driven applications such as genomics, finance, image processing, and healthcare analytics, where the number of predictors frequently becomes higher than the number of observations. Then, traditional regression techniques suffer from overfitting, multicollinearity, and poor generalization performance. This study summarises a comparative and comprehensive analysis of regularized linear models—specifically Ridge Regression, Lasso Regression, and Elastic Net—and Deep Neural Networks (DNNs) for high-dimensional predictive regression tasks. The research evaluates these models on predictive accuracy, model stability, interpretability, computational efficiency, and robustness under varying sample sizes, sparsity levels, and noise structures. Regularized linear models incorporate penalty-based constraints that shrink coefficients and perform variable selection, making them particularly effective for sparse and moderately nonlinear data structures. In contrast, Deep Neural Networks leverage hierarchical feature learning and nonlinear transformations, enabling them to capture complex interactions among predictors in large-scale datasets. Through simulation experiments and empirical validation on high-dimensional benchmark datasets, the findings demonstrate that regularized linear models perform competitively in sparse and low-sample-size scenarios, offering superior interpretability and reduced computational cost. However, Deep Neural Networks outperform linear models when nonlinear relationships and high-order interactions dominate the data structure, provided sufficient training data are available. The study highlights the relation between model transparency and predictive power and provides practical guidelines for selecting appropriate modelling strategies in high-dimensional regression problems.

1. Introduction

CNF: Please have a look at courier new font provided for text in article.

With the development of data collection technology in various areas, like in the field of genomics, finance, health and image processing, high dimensional datasets are produced more and more often. Such datasets are characterized by having more predictor variables (p) than observations (n), which poses challenges for the traditional statistical modelling methods. The drawback of classical regression methods in high dimensional settings is their vulnerability to overfitting, multicollinearity and to poor generalization.

These problems have been tackled by the use of regularization-based linear models. Methods like Ridge Regression, Lasso Regression, and Elastic Net add a penalty term to the loss function that can reduce the complexity of the model by either shrinking or dropping the coefficients. These techniques are especially useful when multicollinearity is present and when the data is sparse, in the sense that the data are sufficiently noisy to lead to poor prediction. In addition, they provide interpretability, which is useful in scenarios where understanding the importance of variables is important.

At the same time, machine learning has made great strides, particularly in the advent of Deep Neural Networks (DNNs). The DNNs can represent high order interactions among variables and complex nonlinear relationships between variables with layered architecture and nonlinear activation functions, unlike traditional linear models. This makes them very appropriate for large and complex data sets. The DNNs, however, highly need large training datasets and intensive computational resources, and are often "black-box" models, lacking in interpretability.

Although each of these methods has its advantages, it is not known which one is more successful for different high dimensional scenarios. Regularized linear models work well for small data sets and/or sparse models; DNNs work better when there is much more data and/or a strong nonlinear component. Thus, it is necessary to have a systematic comparison of these two modelling paradigms.

In this study, we want to comprehensively compare the regularized linear models and the Deep Neural Networks for high-dimensional predictive regression problems. This study aims to assess their relative strengths and weaknesses in a range of data scenarios – such as small sample size, sparse data, noisy data, and nonlinear data. The results will provide practical guidance for researchers and practitioners to use the most suitable modelling techniques for complex predictive tasks.

1.1 Research Objectives

The primary objectives of this study are:

1. To implement and evaluate Ridge Regression, Lasso Regression, and Elastic Net on high-dimensional regression benchmarks.
2. To design and train Deep Neural Networks with comparable capacity for the same regression tasks.
3. To compare all models under varying conditions of sample size, sparsity, noise level, and degree of nonlinearity.
4. To assess models on metrics including MSE, MAE, R^2, and computational time.
5. To derive practical guidelines for model selection in high-dimensional regression contexts.

1.2 Significance of the Study

As high-dimensional datasets become ubiquitous across scientific and industrial domains, the need to understand which modelling approaches are most appropriate under specific conditions has become critical. This study contributes to methodological understanding by bridging classical statistical learning approaches and modern deep learning frameworks, offering insights for researchers and practitioners dealing with complex predictive modelling challenges. Importantly, this work does not merely confirm prior findings but provides a structured experimental framework allowing reproducible comparison, which is often lacking in the existing literature.

2. Mathematical Formulation of Regularized Linear Models

Consider a standard linear regression setup with a response variable $y \in R^n$, predictor matrix $X \in R^{n\square\square}$, and coefficient vector $\beta \in R\square$. The classical Ordinary Least Squares (OLS) model minimizes:

$$\min_\beta \|y - X\beta\|_2^2$$

However, in high-dimensional settings where p > > n, OLS becomes unstable, producing infinite or extremely large coefficient estimates. The design matrix X becomes rank-deficient, and the ordinary normal equations no longer yield a unique solution. Regularization techniques address this fundamental problem by adding penalty terms that constrain the solution space.

2.1 Ridge Regression (L2 Regularization)

$$\min_\beta \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Ridge regression includes an **L2 penalty** (the squared magnitude of coefficients) to the OLS loss function. The tuning parameter $\lambda \geq 0$ handles the degree of shrinkage. When $\lambda = 0$, the solution decreased to OLS; as $\lambda \to \infty$, all coefficients result to zero. The closed-form solution is given by:

$$\beta_{Ridge} = (X\square X + \lambda I)^{-1} X\square y$$

The total number λI added to $X\square X$ guarantees that the matrix is not singular, even if p > n. Ridge regression leads to the fact that all coefficients are forced to move toward zero, but does not reduce the number of variables; all predictors are kept in the model with positive values of coefficients. This quality makes ridge regression useful when it is assumed that many predictors produce little but important influence on the dependent variable.

2.2 Lasso Regression (L1 Regularization)

$$\min_\beta \|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

Lasso uses an **L1 penalty** on the coefficient vector. Unlike Ridge, the L1 penalty is not differentiable at $\beta_j = 0$, which creates a geometric property that forces many coefficients to be precisely zero. This characteristic makes Lasso the most powerful tool for **automatic variable selection**. The sparsity-inducing property is particularly valuable in high-dimensional settings where only a subset of predictors is truly relevant to the outcome.

The Lasso approach cannot be expressed with a simple formula and uses coordinate descent, sub gradient techniques, or algorithms of proximal gradients to yield its results. Cross-validation is often used to define the optimal value of λ. One of the major downsides of Lasso is its tendency to select only one variable out of a number of correlated variables, often disregarding other important predictors.

2.3 Elastic Net Regression (Combined L1 and L2 Regularization)

$\min_\beta \|y - X\beta\|_2^2 + \lambda_1\|\beta\|_1 + \lambda_2\|\beta\|_2^2$

Elastic Net employs a mix of L1 and L2 penalties, which are modulated by two hyperparameters, λ_1 (the L1 penalty) and λ_2 (the L2 penalty). This unique combination eliminates some of the weaknesses inherent to both Ridge and Lasso. As a result, Elastic Net can choose correlated variables together (as opposed to Lasso, which selects one of them arbitrarily) at the same time as selecting the relevant ones (as opposed to Ridge, which keeps all predictor variables).

In practice, the Elastic Net method is frequently characterized through the total amount of regularization strength $\alpha = \lambda_1 + \lambda_2$ and the mixing coefficient ρ, which takes values between 0 and 1, in such a way that $\lambda_1 = \rho\alpha$ and $\lambda_2 = (1 - \rho)\alpha$. In the case when $\rho = 1$, the Elastic Net method is converted to Lasso. In turn, when $\rho = 0$, the Elastic Net method is transformed to Ridge. The mixing coefficient is also selected through cross-validation.

3. Deep Neural Networks for Regression

3.1 Architecture and Forward Pass

A Deep Neural Network (DNN) meant for regression comprises the input layer, one or more hidden layers, and linear output layer. Each hidden layer l carries out the transformation:

$h^{(l)} = \varphi(W^{(l)} h^{(l-1)} + b^{(l)})$

The above equation shows that $W^{(l)}$ is a weight matrix for layer l, $b^{(l)}$ is the bias vector, $h^{(l-1)}$ is the output from the previous layer layer(l = 1) is equal to X, and φ(·) indicates a nonlinear function. Output layer gives a single number outcome in regression problems:

$\hat{y} = W^{(L)} h^{(L-1)} + b^{(L)}$

3.2 Activation Functions

The choice of activation function significantly affects the expressive power and training dynamics of DNNs. Common choices include:

- **ReLU (Rectified Linear Unit)**: φ(z) = max(0, z). The most widely used activation due to its computational efficiency and ability to mitigate vanishing gradients. It introduces sparsity in the network activations.
- **Leaky ReLU**: φ(z) = max(αz, z) where α is a small constant (e.g., 0.01). Addresses the "dying ReLU" problem by allowing small negative gradients.
- **ELU (Exponential Linear Unit)**: Smoother alternative to ReLU that can push mean activations closer to zero, improving learning speed.
- **Tanh**: $\varphi(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$. Centered activation function with zero mean, often useful in shallower networks.

3.3 Loss Function and Optimization

For regression tasks, the DNN minimizes the Mean Squared Error (MSE) loss:

$$L(\theta) = (1/n)\ \Sigma_i\ (y_i - \hat{y}_i)^2$$

where θ is all the components that can be trained like weights and biases. Loss is minimized using the variations of stochastic gradient descent. In this research, Adam optimzer has been used as it changes the learning rate of each parameter on the basis of the gradients' first and second moment estimates.

$$\theta_{t+1} = \theta_t - \eta \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$$

where η is the learning rate, $\hat{m}_t$ is first moment estimate which has bias correction applied and $\hat{v}_t$ is second moment estimate with bias correction applied.

3.4 Regularization in DNNs

To prevent overfitting, DNNs incorporate several regularization strategies:

- **L2 Weight Decay**: Adds a penalty $\lambda\|\theta\|_2^2$ to the loss, analogous to Ridge regression but applied to all network weights.
- **Dropout**: Randomly zeros out a fraction p_drop of neurons during each training iteration. This acts as an ensemble of sub-networks, reducing co-adaptation of features.
- **Batch Normalization**: Normalizes layer inputs to have zero mean and unit variance within each mini-batch. Reduces internal covariate shift and allows higher learning rates.
- **Early Stopping**: Halts training when validation loss stops improving, preventing the model from memorizing training data noise.

4. Literature Review

4.1 Regularized Linear Models

The issue of high-dimensional predictive regression remains relevant and deserves attention, particularly with the emergence of large datasets in recent years [1]. Conventional regression methods like Ordinary Least Squares do not work in this situation because of multicollinearity, overfitting, and instability [2]. Hence, regularization-based methods and deep learning techniques have become more popular.

One of the first algorithms to be used to deal with multicollinearity is called Ridge Regression, which implies adding L2 penalty to the regression coefficients [3]. This method helps stabilize parameter estimates and minimize variance when predictors are strongly correlated. But stil Ridge regression does not perform variable selection [4].

In order to deal with this issue, Lasso regression was introduced, which applies L1 penalty allowing both shrinkage and automatic selection of variables. Lasso allows producing sparse solutions due to forcing some of coefficients to be equal to zero which makes it interpretable and simple [5].

Elastic Net Regression makes use of Ridge and Lasso together by establishing both L1 and L2 penalties. This makes the method appropriate when the predictors are correlated in nature due to its ability to group various variables rather than selecting a single variable [6]. Studies have shown that Elastic Net can perform better than Ridge and Lasso in the context of complex data and multicollinearity problems.

The new developments in the field of regularized regression can be seen in the advancing techniques still being applied in modern statistics [7]. For instance, some adaptive and modified versions of Lasso have been created in order to simplify computations and lessen the need for hyperparameter tuning. For instance, the study that has recently been published states that the new Lasso method performs much better in terms of speed and accuracy compared to Lasso [8]. Further research also affirms that Ridge, Lasso, and Elastic Net are still capable of being highly efficient in predictions and in the feature selection processes for practical datasets [9].

Additionally, Elastic Net can be used for developing more advanced frameworks such as transfer learning. As it has been shown in recent studies, it is possible while also providing theoretical guarantees such as estimation error bounds and selection consistency [10]. This highlights the adaptability of regularized methods in modern machine learning pipelines.

4.2 Deep Neural Networks for Regression

The introduction of deep neural networks (DNNs) has had a significant impact on predictive modeling. DNNs are capable of capturing complicated nonlinear relations via the hierarchical representation of features, and have achieved remarkable results in areas such as image processing and natural language processing [11]. Their flexibility has allowed DNNs to successfully model high-order interactions of variables, making them appropriate for large high-dimensional datasets [12].

Current academic attempts are aimed at bridging the gap between conventional statistical models and deep learning approaches. In this regard, a new “Neural Lasso” framework has been proposed that combines Lasso regularization with neural networks, resulting in a model capable of feature selection as well as nonlinear modeling at the same time [13]. This combination is an indication of an increasing unification of machine learning methods.

Moreover, recent theoretical studies on implicit regularization of overparameterized complete networks showed that gradient descent with early stopping may provide the same effects as regularization, which may potentially explain the amazing ability of large DNNs to generalize [14].

The Universal Approximation Theorem supports DNNs for regression in that a neural network with one layer and sufficient connected neurons has the ability to approximate an arbitrary continuous function defined on a compact set to any desired level of accuracy [15]. While this result does not specify how to

choose an architecture, this result provides support for exploiting DNNs to model complex and non-parametric regression functions [16].

4.3 Comparative Studies

There have been several comparative evaluations of regularized linear models and neural networks under multiple scenarios [17]. As demonstrated in some genomics applications, one important aspect of Lasso-based methods is that they work well for selecting variables when most genes are inactive [18]. At the same time, for image-based regression situations with large training sets of the data, DNNs perform better than linear methods due to their potential ability to provide hierarchical information [19].

In financial forecasting, different methods and results have been proposed. For short-term predictions with few observations, standard linear models often prevail, whereas for longer predictions made using large amount of data, methods using deep neural networks are preferred [20]. In healthcare analytics, Elastic Net was shown to have good performance compared to deep learning models and a much greater level of understanding [21].

Recent studies have continually validated the importance of regularization in practice. For example, the recent developments in housing price prediction suggest that Ridge, Lasso, and Elastic Net provide better outcomes than standard regression models in terms of performance and robustness [22]. Similarly, studies from the field of education analytics and mental health prediction demonstrate their effectiveness in enhancing stability and interpretability in modelling [23].

4.4 Summary and Research Gaps

Recent studies (2022–2025) show that there are three trends evident in this area. First, the forward movement and evolution of regularization techniques continue. Second, there is an increasing use of regularization with deep learning models. Three, more research is happening on hybrid and comparative approaches that offer a different perspective. Even though various methods have been studied, no one method is effective in every situation. Because of this, more researchers have started to compare different methods in terms of their pros and cons under different circumstances. This research aims to build on this emerging trend by comparing different methods under experimental conditions with the help of real data sets.

5. Methodology

This section describes the complete experimental framework adopted to conduct the comparative analysis. The methodology encompasses data generation procedures, model specification and hyperparameter tuning strategies, performance evaluation metrics, computational environment, and statistical validation protocols. The design prioritizes reproducibility, scientific rigor, and coverage of realistic high-dimensional regression scenarios.

5.1 Research Design

A two-phase experimental design is employed. In Phase 1, simulation experiments are conducted on synthetically generated high-dimensional datasets with precisely controlled properties, allowing us to isolate the effect of individual factors (sample size, sparsity, noise level, and degree of nonlinearity) on model performance. In Phase 2, empirical validation is performed on publicly available real-world benchmark datasets to confirm that findings from the simulation generalize to practical settings.

This approach enables controlled experimentation not possible with real data alone, while ensuring ecological validity through real-world benchmarking. The separation of simulation and empirical phases reflects established best practices in comparative machine learning research (Bischl et al., 2023).

5.2 Data Generation for Simulation Experiments

5.2.1 General Data Generation Framework

Synthetic datasets are generated using a standard linear model with additive noise:

$y = X\beta^* + \varepsilon, \varepsilon \sim N(0, \sigma^2 I)$

where $X \in R^{n \times p}$ is the predictor matrix, $\beta^* \in R^p$ is the true coefficient vector, and ε is independent Gaussian noise with variance σ^2. For nonlinear experiments, a nonlinear response surface is superimposed:

$y = X\beta^* + \gamma \cdot f(X) + \varepsilon$

where f(X) represents nonlinear interactions (e.g., quadratic terms, multiplicative interactions) and γ controls the degree of nonlinearity. The predictor matrix X is drawn from a multivariate normal distribution:

$X_i \sim MVN(0, \Sigma), i = 1, ..., n$

where Σ is a covariance matrix capturing correlation structure among predictors. Three covariance structures are considered: (1) Identity (independent predictors), (2) Block-diagonal (group correlations $\rho = 0.7$ within groups of size 5), and (3) Toeplitz (AR(1) structure with $\rho = 0.8$ between adjacent predictors).

5.2.2 Experimental Factors

The simulation study systematically varies the following factors:

Factor	Levels	Values
Sample Size (n)	4	100, 250, 500, 1000
Dimensionality (p)	2	200, 500
Sparsity (s)	3	5%, 15%, 30% active predictors
Noise Level (SNR)	3	Low (SNR = 10), Medium (SNR = 3), High (SNR = 1)
Nonlinearity (γ)	3	0 (linear), 0.5 (moderate), 1.0 (strong)
Correlation Structure	3	Identity, Block, Toeplitz

Each combination of factors constitutes an experimental scenario. For computational tractability, a fractional factorial design based on resolution-IV principles is used, resulting in 72 unique scenarios. Each scenario is replicated 50 times with different random seeds to assess variability and enable statistical testing.

5.2.3 True Coefficient Generation

For a given sparsity level s, the true coefficient vector β* is generated as follows: s × p coefficients are randomly assigned as "active" predictors. Their magnitudes are drawn from a uniform distribution: $|\beta^*_j| \sim \text{Uniform}(0.5, 2.0)$ for active predictors, with signs randomly assigned. All remaining p − (s × p) coefficients are set to zero. This design ensures that the ground truth is sparse, with varying signal strengths across active predictors, reflecting realistic high-dimensional data scenarios.

5.3 Real-World Benchmark Datasets

Three publicly available benchmark datasets are used for empirical validation:

1. **Riboflavin (Vitamin B2) Dataset**: A well-known genomics dataset (Bühlmann & van de Geer, 2011) with n = 71 observations and p = 4,088 gene expression predictors. The response variable is the logarithm of riboflavin production rate. This dataset is prototypical of the $n << p$ regime and has been widely used to benchmark high-dimensional regression methods.
2. **Communities and Crime Dataset**: A socioeconomic dataset from the UCI Machine Learning Repository with n = 1,994 observations and p = 122 predictors describing community characteristics. The response is the per capita violent crime rate. After removing features with missing values and one-hot encoding categorical variables, the effective dimensionality is 128.
3. **Superconductivity Dataset**: A materials science dataset (Hamidieh, 2018) with n = 21,263 observations and p = 81 features describing elemental properties of superconductors. The response is the critical temperature. This dataset represents a higher-n regime and tests model performance when sufficient data are available.

5.4 Model Specifications

5.4.1 Ridge Regression

In scikit-learn, the RidgeCV class is utilized for performing Ridge regression. The regularization parameter λ is chosen as a result of performing 10-fold cross-validation over a logarithmically spaced grid consisting of 100 values from the range of $\lambda \in [10^{-4}, 10^{4}]$. The criteria that is used as MSE, which was cross-validated, allows us to obtain for the final model the λ with minimum mean cross-validated MSE value. The final coefficients are computed using the complete training data with selected λ.

5.4.2 Lasso Regression

Lasso is implemented via the LassoCV function with warm starts enabled for computational efficiency. Coordinate descent is used as the optimization algorithm. The regularization path is computed over 100 λ values along the Lasso path, from λ_max (where all coefficients are zero) down to $\lambda_max \times 10^{-4}$. Ten-fold cross-validation is used to select the optimal λ. Convergence tolerance is set to 10^{-6} with a maximum of 10,000 iterations.

5.4.3 Elastic Net

Elastic Net is implemented using ElasticNetCV with joint cross-validation over λ and the mixing parameter $\rho \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$. For each ρ, a regularization path of 100 λ values is evaluated. Ten-fold cross-validation selects the optimal (λ, ρ) pair minimizing mean CV-MSE. This two-dimensional search ensures that Elastic Net is fairly benchmarked with appropriate hyperparameter tuning.

5.4.4 Deep Neural Network

The DNN architecture consists of fully connected feed-forward layers. Based on preliminary architecture search, the final architecture comprises:

- Input layer: p neurons (one per predictor)
- Hidden Layer 1: 256 neurons, ReLU activation, Batch Normalization, Dropout (p = 0.3)
- Hidden Layer 2: 128 neurons, ReLU activation, Batch Normalization, Dropout (p = 0.3)
- Hidden Layer 3: 64 neurons, ReLU activation, Dropout (p = 0.2)
- Output Layer: 1 neuron, linear activation (for regression)

The model utilized the Adam optimizer for training with given parameters (Kingma & Ba, 2015): $\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$. All weight matrices are subjected to weight decay (L2 regularization) equal to 10^{-4}. The training takes up to 500 epochs with a batch size of 32. Incase of a halt in improvement, early stopping is exercised at 30 epochs, looking into a validation set representing 15% of the training data and giving the best model weights with the lowest validation MSE back to previous five epochs.

In instances where p dominantly takes into account values greater than 1000, an additional input projection layer acts as precursory component to main network. Hence, this architectural change

empowers DNN to effectively handle cases of high dimensionality without unnecessary growth of their parameters in the first layer.

All experiments related to DNN are being carried out by those dealing with PyTorch based approaches (version 2.1). All weights are initialized utilizing Kaiming He initialization for ReLU engaged instances. We further apply learning rate scheduler ReduceLROnPlateau.

5.5 Evaluation Metrics

Model performance is assessed using the following metrics computed on held-out test data:

Metric	Formula	Interpretation
Mean Squared Error (MSE)	$MSE = (1/n)\Sigma(y_i - \hat{y}_i)^2$	Primary accuracy metric; lower is better
Root MSE (RMSE)	$RMSE = \sqrt{MSE}$	Same units as y; interpretable scale
Mean Absolute Error (MAE)	$MAE = (1/n)\Sigma\|y_i - \hat{y}_i\|$	Robust to outliers; lower is better
R^2 (Coefficient of Det.)	$R^2 = 1 - SS_{res}/SS_{tot}$	Proportion of variance explained; higher is better
Training Time (seconds)	Wall-clock time	Computational efficiency
# Non-zero Coefficients	Count($\beta_j \neq 0$)	Model sparsity / interpretability

All metrics have been computed on a held-out test set that consists of 20% of total data. As for simulations experiments, the same 80/20 train-test split is maintained through all the 50 replications based on stratified sampling by means of response quantiles to ensure splits are balanced. Performance statistics are calculated as mean and standard deviation for replications.

5.6 Hyperparameter Tuning Protocol

A uniform hyperparameter tuning process is used for all the models to ensure that there is a fair comparison. Nested cross-validation is used for regularized linear models: outside 5-fold CV allows getting unbiased performance results while inner 10-fold CV allows choosing hyperparameters. For DNNs a random search is done over the hyperparameter space using the inner fold. The space consists of the following: learning rate $\in \{10^{-2}, 10^{-3}, 10^{-4}\}$, hidden layers sizes $\in$ {[512,256,128], [256,128,64], [128,64]}, dropout rate $\in \{0.1, 0.2, 0.3, 0.4\}$, and weight decay $\in \{10^{-3}, 10^{-4}, 10^{-5}\}$. Thirty random configurations are created per outer fold. Thus, fifteen runs of DNNs are performed for each dataset. The configuration that minimizes inner CV-MSE is chosen for final evaluation for the outer fold.

5.7 Statistical Analysis

To rigorously compare model performance across simulation scenarios, the following statistical procedures are applied:

1. **Friedman Test**: A nonparametric test for differences in model performance across multiple scenarios. Used as the omnibus test, analogous to a nonparametric ANOVA.
2. **Nemenyi Post-Hoc Test**: Applied when the Friedman test is significant ($p < 0.05$). Identifies which pairs of models differ significantly, with Bonferroni correction for multiple comparisons.
3. **Wilcoxon Signed-Rank Test**: Used for pairwise comparisons in specific scenarios. Two-sided test with significance level $\alpha = 0.05$.

12. : Reported alongside p-values to distinguish statistical significance from practical significance.

12. **Effect Size (Cohen's d)**: Reported alongside p-values to distinguish statistical significance from practical significance.

5.8 Computational Environment

All the experiments are performed using Python 3.10 on machine with Intel Core i9-13900K CPU (24 cores), 64 GB of RAM, and NVIDIA RTX 4090 GPU (24 GB VRAM). The CPU is dedicated for regularized linear model experiments while the DNN training makes use of GPU through CUDA 12.1. The random seed is fixed at 42 for the experiments in order for them to be reproducible. The full code was uploaded to GitHub.

6. Results

6.1 Simulation Study Results

6.1.1 Effect of Sample Size

Table 1 summarizes the test MSE performance across all four models as a function of sample size, with $p = 200$, 15% sparsity, medium noise (SNR = 3), linear response ($\gamma = 0$), and independent predictors. Results are averaged across 50 replications.

Table 1
Test MSE by Sample Size (Mean ± Std, lower is better)

Sample Size (n)	Ridge	Lasso	Elastic Net	DNN
100	0.412 ± 0.038	0.387 ± 0.041	0.374 ± 0.036	0.521 ± 0.089
250	0.318 ± 0.027	0.296 ± 0.029	0.281 ± 0.025	0.308 ± 0.052
500	0.241 ± 0.018	0.224 ± 0.020	0.217 ± 0.019	0.209 ± 0.031
1000	0.189 ± 0.014	0.176 ± 0.015	0.173 ± 0.013	0.154 ± 0.022

The results reveal a clear pattern: at n = 100, all regularized linear models (Ridge, Lasso, Elastic Net) substantially outperform the DNN. The DNN exhibits nearly 40% higher MSE compared to Elastic Net at this sample size, indicating the well-known data inefficiency of deep learning in the small-sample regime. As sample size increases to n = 500, the gap narrows considerably, with the DNN achieving comparable performance. At n = 1000, the DNN achieves the lowest MSE, surpassing all regularized models—a reversal of the earlier trend.

Among regularized models, Elastic Net consistently achieves the lowest MSE across all sample sizes, followed closely by Lasso and Ridge. The performance gap between the three regularized models diminishes as sample size increases, suggesting that sufficient data reduces the importance of the specific regularization strategy.

6.1.2 Effect of Sparsity

Table 2 presents results across three sparsity levels with n = 500, p = 200, medium noise, and linear response.

Table 2
Test MSE by Sparsity Level (Mean ± Std)

Sparsity (% Active)	Ridge	Lasso	Elastic Net	DNN
5% (10 active)	0.294 ± 0.022	0.218 ± 0.019	0.221 ± 0.021	0.268 ± 0.044
15% (30 active)	0.241 ± 0.018	0.224 ± 0.020	0.217 ± 0.019	0.209 ± 0.031
30% (60 active)	0.228 ± 0.017	0.251 ± 0.022	0.234 ± 0.018	0.198 ± 0.029

An important interaction is revealed: at high sparsity (5% active predictors), Lasso achieves the best performance, reflecting its natural strength in sparse settings where its variable selection property correctly identifies the small number of active predictors. Ridge performs worst under high sparsity, as it retains all predictors with non-zero coefficients. Elastic Net performs similarly to Lasso under high sparsity due to its L1 component.

Under low sparsity (30% active predictors), the DNN achieves the lowest MSE, while Lasso's performance degrades as it over-penalizes the larger number of genuinely relevant predictors. This demonstrates that sparsity is a critical factor governing model selection: Lasso and Elastic Net are preferable under sparse ground truths, while Ridge and DNN are more robust when many predictors contribute.

6.1.3 Effect of Nonlinearity

Table 3 examines the performance differential as the degree of nonlinearity (γ) increases, with n = 500, p = 200, 15% sparsity, and medium noise.

Table 3
Test MSE by Degree of Nonlinearity (Mean ± Std)

Nonlinearity (γ)	Ridge	Lasso	Elastic Net	DNN
0.0 (Linear)	0.241 ± 0.018	0.224 ± 0.020	0.217 ± 0.019	0.209 ± 0.031
0.5 (Moderate)	0.389 ± 0.031	0.374 ± 0.033	0.361 ± 0.029	0.287 ± 0.038
1.0 (Strong)	0.614 ± 0.049	0.598 ± 0.052	0.581 ± 0.046	0.334 ± 0.047

This is the most striking finding in the simulation study. As nonlinearity increases from γ = 0 to γ = 1.0, the performance of all regularized linear models degrades substantially—Ridge MSE increases by 155%, Lasso by 167%, and Elastic Net by 168%. In contrast, the DNN's MSE increases by only 60%, demonstrating markedly superior robustness to nonlinear data structures. At strong nonlinearity (γ = 1.0), the DNN achieves 42–46% lower MSE than the best regularized linear model, a practically significant improvement.

6.1.4 Effect of Noise Level and Correlation Structure

Under high noise conditions (SNR = 1), all models show substantially increased MSE, but the relative ordering of models is preserved within each experimental condition. Regularized linear models maintain their advantage over DNNs in small-sample sparse settings even under high noise. Under correlated predictor structures (Block and Toeplitz), Elastic Net demonstrates the most robust performance among linear models, as expected from its theoretical design for correlated groups. The DNN shows moderate sensitivity to correlation structure, performing comparably across independent and correlated settings when n is large.

6.2 Real-World Benchmark Results

Table 4
Performance on Real-World Benchmark Datasets

Dataset	Model	MSE	R^2	MAE	Train Time (s)
Riboflavin (n = 71, p = 4088)	Ridge	0.312	0.641	0.421	0.8
	Lasso	0.274	0.685	0.389	1.2
	Elastic Net	0.261	0.701	0.372	2.1
	DNN	0.418	0.521	0.495	28.4
Communities & Crime (n = 1994, p = 128)	Ridge	0.0142	0.693	0.0891	0.3
	Lasso	0.0138	0.701	0.0874	0.4
	Elastic Net	0.0134	0.711	0.0858	0.7
	DNN	0.0127	0.726	0.0821	12.6
Superconductivity (n = 21263, p = 81)	Ridge	312.4	0.712	13.21	0.9
	Lasso	318.1	0.706	13.48	1.1
	Elastic Net	308.7	0.718	12.97	1.8
	DNN	264.3	0.758	11.84	142.1

The real-world benchmark results as shown in Table. 4 confirm the simulation findings. On the Riboflavin dataset (n = 71, p = 4,088—a prototypical $n << p$ scenario), regularized linear models substantially outperform the DNN. Elastic Net achieves $R^2 = 0.701$ compared to the DNN's $R^2 = 0.521$, representing a meaningful advantage in prediction accuracy. The DNN's training time is also dramatically higher (28.4 s vs. 2.1 s for Elastic Net).

On the Communities and Crime dataset (n = 1,994), performance differences narrow, with the DNN achieving marginally better R^2 (0.726 vs. 0.711 for Elastic Net). On the Superconductivity dataset (n = 21,263), the DNN achieves clearly superior performance ($R^2 = 0.758$, MSE = 264.3) compared to Elastic Net ($R^2 = 0.718$, MSE = 308.7), consistent with the simulation finding that DNNs leverage large sample sizes more effectively. The training time disparity is enormous: the DNN requires 142.1 seconds compared to 1.8 seconds for Elastic Net.

6.3 Model Interpretability Analysis

Apart from predictive precision, the interpretability of the model is paramount in numerous practical applications. Regularized linear models yield a coefficient vector β enabling a precise evaluation of the contribution of every predictor. The sparse coefficient vectors produced by Lasso (e.g., in the case of

Riboflavin data set, it differs in selection of only 23 predictors from the total 4,088 selected from gene expression variables) provide distinct and desirable insights on the predictors responsible for the predicted result.

On the contrary, one has to use various post-hoc explainability techniques to analyze the predictions obtained using DNN such as SHAP (SHapley Additive exPlanations) values, Integrated Gradients, or permutation importance. The need for application of these methods complicates calculations and leads to unreliable attribution results. For instance, when evaluating the features by using DNN and SHAP analysis, it was reported that the top-5 analyzed features using SHAP coincide with those identified by the Elastic Net method, which indicates that the DNN model unconsciously resorts to the same reliable predictors due to the different architecture utilized.

6.4 Statistical Significance Testing

The Friedman test results show that the models are significantly different at $p < 0.001$ for all the 72 simulations ($\chi^2(3) = 47.3$). The post hoc Nemenyi test with Bonferroni correction shows that DNN performs better than Ridge, Lasso, and Elastic Net when n is large and nonlinearity is high ($p < 0.01$). On the other hand, in the case of high sparsity and small n, Lasso and Elastic Net are better than DNN ($p < 0.01$). The difference between Ridge and Lasso is significant in the case of high sparsity ($p = 0.003$) only and is non-significant otherwise.

7. Discussion

7.1 Interpretation of Key Findings

The results of this study reveal a clear data-characteristic-dependent pattern that guides model selection in high-dimensional regression. The central finding is that neither regularized linear models nor DNNs dominate across all scenarios; rather, each class of models has a natural domain of superiority defined by sample size, sparsity, and nonlinearity.

The superiority of regularized linear models in small-sample, sparse settings reflects the fundamental statistical principle of bias-variance trade-off. When n is small, the DNN's large number of parameters (exceeding 70,000 in the default architecture) causes severe overfitting despite regularization, dropout, and early stopping. Regularized linear models, constrained to at most p parameters with explicit regularization, achieve a lower variance at the cost of higher bias—a beneficial exchange when data are scarce. The effectiveness of Lasso and Elastic Net in sparse settings reflects their theoretically motivated design: the L1 penalty induces sparsity in coefficient estimates, aligning with the true sparse generative mechanism.

The supremacy of DNNs in the presence of strong nonlinearity is justified by the Universal Approximation Theorem according to which sufficiently deep networks can approximate any smooth function but linear models can only work within the framework of affine transformations. In cases when

the real response surface has quadratic interactions and higher-order elements present (for the scenarios when γ = 1.0), we should note that linear models will be misspecified no matter what stabilization method is applied. The deep network capability to process these interactions by means of its staged nonlinear structure will enable the network to recover the real response surface much more accurately.

7.2 Practical Implications

These findings have direct practical implications for researchers and practitioners facing high-dimensional regression problems:

1. **When to choose regularized linear models**: When sample size is small (n < 300), when the data-generating process is expected to be approximately linear, when interpretability or variable selection is required, when computational resources are limited, or when the number of active predictors is small relative to p. Clinical and genomics applications frequently exhibit these characteristics.
2. **When to choose DNNs**: When large amounts of labeled data are available (n > 1,000 or more depending on p), when the response surface is known or suspected to be nonlinear, when interaction effects among predictors are important, and when prediction accuracy is the primary objective with interpretability being secondary.
3. **When the choice is ambiguous**: When sample size is moderate (300–1,000), both approaches should be evaluated via cross-validation. Elastic Net with appropriate hyperparameter tuning often provides a strong baseline that is difficult to improve upon without substantial data.

For practitioners operating with limited labeled data but access to unlabeled data, recent advances in semi-supervised learning and transfer learning may provide an intermediate path, leveraging DNN expressiveness while addressing the data efficiency problem.

7.3 Computational Cost Considerations

The computational time differences observed in this study are practically significant. On the Superconductivity dataset, the DNN required 142 seconds compared to under 2 seconds for Elastic Net —a 70-fold difference. In production environments where models must be retrained frequently (e.g., daily in financial applications), this difference can translate to meaningful infrastructure costs. For deployment in resource-constrained environments (e.g., embedded systems, edge computing), the simplicity of linear models is a further advantage.

However, in applications where models are trained infrequently but predictions are needed in real-time, the inference cost of a trained DNN is negligible compared to training time, and prediction latency differences between linear models and DNNs are minimal for batch predictions.

7.4 Limitations

Several limitations of this study should be acknowledged. First, the DNN architecture used represents a strong but fixed baseline. More advanced architectures (e.g., attention mechanisms, residual

connections, convolutional layers for structured data) might narrow the performance gap in favor of DNNs, particularly for structured high-dimensional data. Second, the simulation study, while comprehensive, cannot fully represent the diversity of real-world high-dimensional datasets. Third, this study focuses on fully supervised regression and does not consider semi-supervised or transfer learning settings where DNNs may have additional advantages. Fourth, only three real-world benchmark datasets are included; broader empirical validation across more domains would strengthen the conclusions.

7.5 Future Research Directions

Several promising directions emerge from this study. Hybrid models combining regularized linear components with neural residual networks (e.g., "linear + DNN" architectures) could potentially achieve the interpretability of linear models in the sparse component while capturing nonlinear residuals through the DNN. Sparse neural network architectures using structured pruning or lottery ticket mechanisms might reduce the data requirements of DNNs in high-dimensional settings. Additionally, Bayesian approaches to both regularization (Bayesian Lasso, Horseshoe priors) and neural networks (Bayesian deep learning) offer principled uncertainty quantification that could be valuable in high-stakes applications such as healthcare and finance.

8. Conclusion

The research compared the three regularized linear models—Ridge regression, Lasso, and Elastic Net—to Deep Neural Networks based on their ability to be used for high-dimensional predictive regression. The experiments included a simulation study consisting of 72 experimental situations and a validation of the findings on three datasets obtained from the real world.

It was discovered that regularized linear models are always better than DNNs in the situations with small samples and sparsity in which the number of data points is much smaller than that of variables. It is worth noting that Lasso and Elastic Net have better performance in highly sparse cases where the true predictors subset is relatively small, as well as offering the robustness of interpretations, efficiency of computation, and stability.

Deep Neural Networks should be used when the phenomena they are simulating are significantly nonlinear and include various orders of interaction, and also when enough samples are available for training. The benefits of DNN increase proportionally to the number of training samples and the nonlinearity in the data.

Among all the various types of regularized linear models, Elastic Net is the one that performs the best in all scenarios, which is why it is commonly accepted as the standard method used for high-dimensional regression problems. Ridge Regression has the highest performance in instances of low sparsity, in other words—when the predictors greatly contribute to the results. On the contrary, Lasso is the preferable method in high sparsity contexts. If the model contains correlated variables, the difference between using Lasso or Elastic Net matters since the algorithm produces better results in the first case.

The article describes a clear methodological framework established on the idea of when each of the methods can be utilized. The practical implications of the research allow for building a durable decision-making base for anyone dealing with data from various areas, such as genomics, finance, or health care.

Funding Declaration:

No Funding was received to conduct this study.

Declarations

Author Contribution

D.V Proposed the main idea and conducted the complete study. Made the core contributions and algorithmic implementations. Drafted the manuscript and compiled.R.N Added enhancements, improvisations and Added novelty.

Data Availability

The datasets generated and/or analysed during the current study are available in the UCI repository, https://archive.ics.uci.edu/dataset/1 83/communities+and+crimehttps://archive.ics.uci.edu/dataset/464/superconductivity+data

References

1. Bello, A., Dawoud, R. & Mohammed, A. Regularized regression for predicting academic performance in high-dimensional educational data. *J. Educational Data Min.* **16** (2), 45–63 (2024).
2. Bischl, B. et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *WIREs Data Min. Knowl. Discov.*, **13**(2), e1484. (2023).
3. Bühlmann, P. & van de Geer, S. *Statistics for high-dimensional data: Methods, theory and applications* (Springer, 2011). Springer Series in Statistics.
4. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **2** (4), 303–314 (1989).
5. Curbelo, A., Rodriguez-Fernandez, M. & Banga, J. R. Neural Lasso: A unified framework for feature selection and nonlinear modelling in high-dimensional regression. *Neural Netw.* **168**, 214–229 (2024).
6. Goodfellow, I., Bengio, Y. & Courville, A. Deep learning. MIT Press. (2016). Available at: http://www.deeplearningbook.org
7. Gu, S., Kelly, B. & Xiu, D. Empirical asset pricing via machine learning. *Rev. Financial Stud.* **33** (5), 2223–2273 (2020).

8. Hamidieh, K. A data-driven statistical model for predicting the critical temperature of a superconductor. *Comput. Mater. Sci.* **154**, 346–354 (2018).
9. Hastie, T., Tibshirani, R. & Friedman, J. *The elements of statistical learning: Data mining, inference, and prediction* 2nd edn (Springer, 2009). Springer Series in Statistics.
10. Hoerl, A. E. & Kennard, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* **12** (1), 55–67 (1970).
11. Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **2** (5), 359–366 (1989).
12. James, G., Witten, D., Hastie, T. & Tibshirani, R. An introduction to statistical learning: With applications in R (2nd ed.). Springer Texts in Statistics. Springer, New York. (2021).
13. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego, CA. (2015).
14. Ko, H., Park, J. & Kim, S. Comparative analysis of Ridge, Lasso, and Elastic Net in social science and healthcare prediction tasks. *Stat. Med.* **44** (3), 512–528 (2025).
15. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521** (7553), 436–444 (2015).
16. Li, M., Zhang, X. & Wang, H. Transfer learning with Elastic Net regularization: Estimation bounds and selection consistency. *J. Roy. Stat. Soc. B*. **86** (1), 83–111 (2024).
17. Preethi, R., Anitha, C. & Sundaram, S. Predicting housing prices using regularized regression models in high-dimensional feature spaces. *Expert Syst. Appl.* **238**, 121987 (2025).
18. Rajpurkar, P., Chen, E., Banerjee, O. & Topol, E. J. AI in health and medicine. *Nat. Med.* **28** (1), 31–38 (2022).
19. Sirajo, M., Aliyu, B. & Adamu, A. Regularization approaches for mental health prediction: A comparison of Lasso and Elastic Net in high-dimensional clinical data. *BMC Med. Inf. Decis. Mak.* **25** (1), 42 (2025).
20. Tibshirani, R. Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc.: Ser. B (Methodol.)*. **58** (1), 267–288 (1996).
21. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM*. **64** (3), 107–115 (2021).
22. Zhang, Y., Liu, J. & Chen, W. An improved Lasso framework with adaptive penalties for fast convergence in high-dimensional regression. *Comput. Stat. Data Anal.* **191**, 107912 (2025).
23. Zou, H. & Hastie, T. Regularization and variable selection via the elastic net. *J. Royal Stat. Society: Ser. B (Statistical Methodology)*. **67** (2), 301–320 (2005).