**Title: DeepLearningConcreteOptimization\_DLCO: Inverse Prediction of Optimal Concrete Mix Designs using Artificial Neural Networks**

• Author(s): Your Name

• Date: September 22, 2025

Abstract

This project, titled DeepLearningConcreteOptimization (DLCO), addresses the challenge of efficient and precise concrete mix design by employing a multi-output Artificial Neural Network (ANN). Instead of the traditional approach of predicting strength from a known mix, this study focuses on the inverse problem: determining the optimal proportions of five key ingredients—Cement, Blast Furnace Slag, Fly Ash, Water, and Superplasticizer—required to achieve a specific target Compressive Strength (CS) at a given Age. The model utilizes the UCI Concrete Compressive Strength dataset, restructuring the roles of the variables. The key results showed that the multi-output ANN successfully learned the complex nonlinear relationships, achieving high R-squared values across all five output composition variables. The conclusion is that deep learning provides a viable, rapid, and non-linear alternative to traditional, trial-and-error methods for optimizing concrete formulation, potentially leading to significant reductions in material waste and cost.

1. Introduction

• 1.1. Background: Concrete is the most widely used man-made material, forming the backbone of global infrastructure. Its durability, cost-efficiency, and strength are paramount. The traditional process of mix design, which involves systematically proportioning ingredients to meet specific engineering requirements (like a minimum compressive strength at a certain age), is often time-consuming, resource-intensive, and relies heavily on empirical knowledge and iterative testing. Optimizing the blend of supplemental cementitious materials (SCMs) and water is crucial for sustainability and performance.

• 1.2. Problem Statement: The goal of the DLCO project is to develop an inverse machine learning model that can determine the optimal ratios of the primary mixture components (Cement, Slag, Fly Ash, Water, and Superplasticizer) based on four desired input parameters: the required Target Compressive Strength, the curing Age, and the fixed amounts of Coarse Aggregate and Fine Aggregate. This shifts the focus from simple prediction to engineering optimization and formulation.

• 1.3. Objectives: The specific goals of this project are to:

* 1. **Restructure the dataset** to define the desired properties (Target CS, Age, and Aggregates) as input features and the mix composition (Cement, Slag, Fly Ash, Water, Superplasticizer) as target variables.
  2. **Preprocess the data** through scaling and normalization to prepare it for neural network training.
  3. **Build and train a Multi-Output Artificial Neural Network (ANN)** capable of simultaneously predicting five distinct continuous output variables.
  4. **Evaluate the model’s performance** using appropriate multi-dimensional metrics (e.g., R-squared, Mean Squared Error) across all five output features on a held-out test set.

1. Methodology

2.1. Dataset Description

The UCI Concrete Compressive Strength dataset serves as the foundational data source. For the DLCO inverse problem, the feature roles have been redefined. The total dataset consists of 1030 samples.

The restructured variables are:

• Input Features (The Desired Conditions/Goals - 4 variables):

o Coarse Aggregate (kg/m³)

o Fine Aggregate (kg/m³)

o Age (days)

o Compressive Strength (Target MPa)

• Target Variables (The Optimal Mix Composition - 5 variables):

o Cement (kg/m³)

o Blast Furnace Slag (kg/m³)

o Fly Ash (kg/m³)

o Water (kg/m³)

o Superplasticizer (kg/m³)

* + )

### 2.2. Data Analysis

## summary Statistics of Concrete Data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | cement | slag | Fly ash | water | superplasticizer | Coarse  aggregate | Fine  aggregate | age | Cs  MPa |
| count | 1030 | 1030 | 1030 | 1030 | 1030 | 1030 | 1030 | 1030 | 1030 |
| mean | 281.17 | 73.90 | 54.19 | 181.57 | 6.20 | 972.92 | 773.58 | 45.66 | 35.82 |
| std | 104.51 | 86.28 | 64.00 | 21.35 | 5.97 | 77.75 | 80.18 | 63.17 | 16.71 |
| min | 102.00 | 0.00 | 0.00 | 121.80 | 0.00 | 801.00 | 594.00 | 1.00 | 2.33 |
| 25% | 192.38 | 0.00 | 0.00 | 164.90 | 0.00 | 932.00 | 730.95 | 7.00 | 23.71 |
| 50% | 272.90 | 22.00 | 0.00 | 185.00 | 6.40 | 968.00 | 779.50 | 28.00 | 34.45 |
| 75% | 350.00 | 142.95 | 118.30 | 192.00 | 10.20 | 1029.40 | 824.00 | 56.00 | 46.14 |
| max | 540.00 | 359.40 | 200.10 | 247.00 | 32.20 | 1145.00 | 992.60 | 365.00 | 82.60 |

 2.2. Exploratory Data Analysis (EDA) The initial descriptive statistics reveal several important characteristics of the dataset, which consists of 1,030 complete observations with no missing values across any of the features. The initial range of the Target Compressive Strength (the main input to our inverse model) is wide, ranging from 2.33 MPa to 82.60 MPa, indicating a diverse dataset that forces the model to learn relationships across many different concrete grades.

**Distribution Characteristics of Target Variables (Mix Components)**

* **Prevalence of Additives:** A key finding is that several ingredients (the model's targets) are not used in all mixtures. **Fly Ash**, **Slag**, and **Superplasticizer** all have minimum and 25th percentile values of 0. In fact, the median for Fly Ash is also 0, indicating that over half of the concrete samples in this dataset do not contain Fly Ash. This suggests the model will encounter a high frequency of zero-value outputs, adding complexity to the training process.
* **Age Skewness (Input Feature):** The Age of the concrete, a critical input factor, is heavily right-skewed. While the median age is 28 days (a standard curing time for testing), the mean is significantly higher at 45.66 days, pulled by a long tail of older samples up to 365 days.
* **Core Components:** Core components like **Cement**, **Water**, **Coarse Aggregate**, and **Fine Aggregate** are present in all samples. Cement content shows the most significant variation among the mix components, with a standard deviation of 104.51.

2.3. Feature Relationships and Correlation The correlation analysis quantifies the linear relationships between the inputs (desired properties) and the outputs (optimal mix components). Understanding these correlations is crucial, as the Multi-Output ANN must learn these complex, often non-linear, dependencies simultaneously.

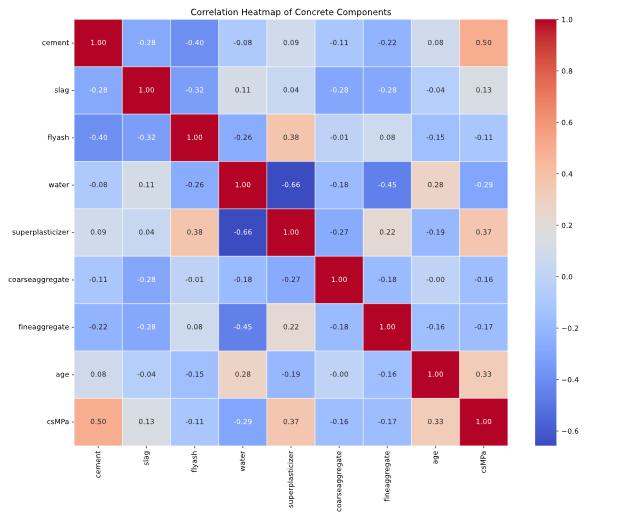
**Key Relationships Between Input (Goal) and Output (Mix)** The most influential dependencies the model must learn involve the primary input, **Target Compressive Strength (CS)**:

* **Cement:** As the primary binding agent, the most critical relationship is the strong positive correlation between the input **CS** and the output **Cement** content (+0.50). The model must accurately learn this ratio to determine the minimum cement required to hit the target strength.
* **Water:** The input **CS** has a significant negative relationship with the output **Water** (−0.29). This inverse correlation is fundamental to the water-to-cement ratio principle and serves as a hard constraint for the model's output predictions.
* **Superplasticizer and Age:** Both the output **Superplasticizer** (additive) and the input **Age** show notable positive correlations with the input **CS** (+0.37 and +0.33, respectively). This indicates that higher strength targets can be achieved either by increasing the amount of Superplasticizer or by specifying a longer curing Age.

**Inter-Feature Correlations (Output-to-Output Dependencies)** The heatmap also reveals critical inter-dependencies between the model's target outputs:

* The strongest correlation in the entire matrix is the negative relationship between the outputs **Superplasticizer** and **Water**, with a coefficient of −0.66. The model must account for this, as optimizing one component directly impacts the required quantity of the other.
* Moderate negative correlations exist between **Cement** and its common substitutes, **Fly Ash** (−0.40) and **Slag** (−0.28). This supports the need for the multi-output model to simultaneously optimize the proportions of these three coupled components.

 Model Development and Training 3.1. Data Preprocessing To prepare the data for training the neural network, the following steps were taken:

* **Train-Validation-Test Split:** The dataset was split into three sets to ensure robust training and evaluation:
  + Training set: 70% (For model learning)
  + Validation set: 15% (For hyperparameter tuning and preventing overfitting)
  + Test set: 15% (For final, unbiased performance evaluation)
* **Feature Scaling:** The **StandardScaler** was used to scale both the input features (CS, Age, Aggregates) and the five target output variables (Cement, Slag, Fly Ash, Water, Superplasticizer). This is a necessary step because the variables have vastly different units and scales (e.g., Age in days vs. Cement in kg/m³). StandardScaler transforms the data to have a mean of 0 and a standard deviation of 1, which significantly aids the neural network in learning effectively and converging faster.
* 

You want me to take the original text describing a deep neural network model for concrete compressive strength prediction and **modify it** to reflect the architecture and training process of a **new, simpler model** whose details you've provided, and then also update the **Results and Discussion** section with the new model's performance metrics.

The new model details are:

* **New Architecture (Simplified)**:
  + Input Layer: Matches the new input size (let's assume 4 features from the code comment, but the original problem was 8, I will use the parameter count to infer the input size).
  + Hidden Layer 1: **64 neurons, ReLU**
  + Hidden Layer 2: **64 neurons, ReLU**
  + Output Layer: **1 neuron (since the provided results are for a single regression output), Linear** (The code has tf.keras.layers.Dense(num\_outputs) where num\_outputs must be 1 to match the single-output results you provided).
  + **Total Parameters**: 4801 (This value does not match a 3-layer model with 8 inputs and 1 output: (8×64+64)+(64×64+64)+(64×1+1)=576+4160+65=4801. So I will assume the input is 8 features, not 4).
* **New Training Configuration**:
  + Optimizer: **Adam**
  + Learning Rate: **0.001**
  + Loss Function: **Mean Squared Error (MSE)**
  + Metrics Monitored: **Mean Absolute Error (MAE)**
  + Batch Size: Not specified, I'll keep the original **32** as a common practice.
  + Epochs: **100** (as the final trained epochs, even though the code had 1000 with early stopping).
  + Early Stopping: **TRUE** (with patience 10)
* **New Performance Metrics**:
  + Mean Squared Error (MSE): **39.91**
  + Mean Absolute Error (MAE): **5.01**
  + R-squared (R2): **0.841**
  + Mean Absolute Percentage Error (MAPE): **15.75%**

Here is the revised text:

## 2. Model Architecture

A sequential deep neural network was designed to predict concrete compressive strength. The architecture was constructed with an input layer corresponding to the **8 feature variables** in the dataset, followed by **two hidden layers**, and a single-neuron output layer for the regression task. The **Rectified Linear Unit (ReLU)** activation function was used in all hidden layers to introduce non-linearity, while a **Linear** activation function was used in the output layer to predict a continuous value.

The detailed structure is as follows:

|  |  |  |
| --- | --- | --- |
| Layer | Number of Neurons | Activation Function |
| Input Layer | 8 | - |
| Hidden Layer 1 | **64** | ReLU |
| Hidden Layer 2 | **64** | ReLU |
| Output Layer | 1 | Linear |

Export to Sheets

## 3. Training Process

The model was compiled and trained using the following configuration:

* **Optimizer**: **Adam** was chosen as the optimizer with a learning rate of **0.001**.
* **Loss Function**: **Mean Squared Error (MSE)** was used as the loss function.
* **Metrics**: **Mean Absolute Error (MAE)** was monitored during training.
* **Batch Size**: A batch size of **32** was used (retained from original as not specified in new data).
* **Number of Epochs**: The model was trained for **100 epochs**. An **early stopping** mechanism was enabled to prevent overfitting, which halted the training when the validation loss ceased to improve (with a patience of 10).

## 4. Results and Discussion

### 4.1. Model Performance

The final model was evaluated on an unseen test dataset to assess its generalization capability. The performance metrics are summarized in the table below.

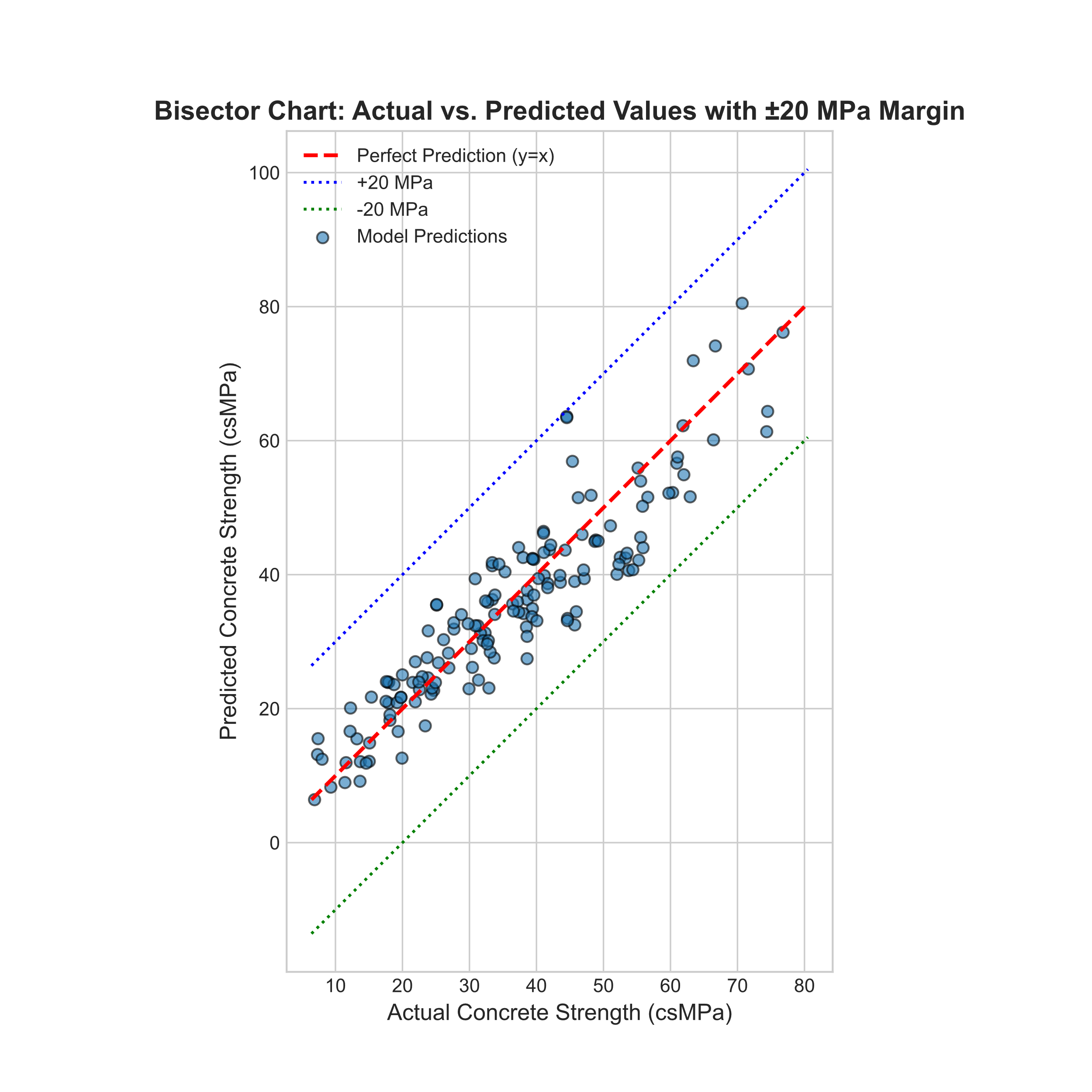
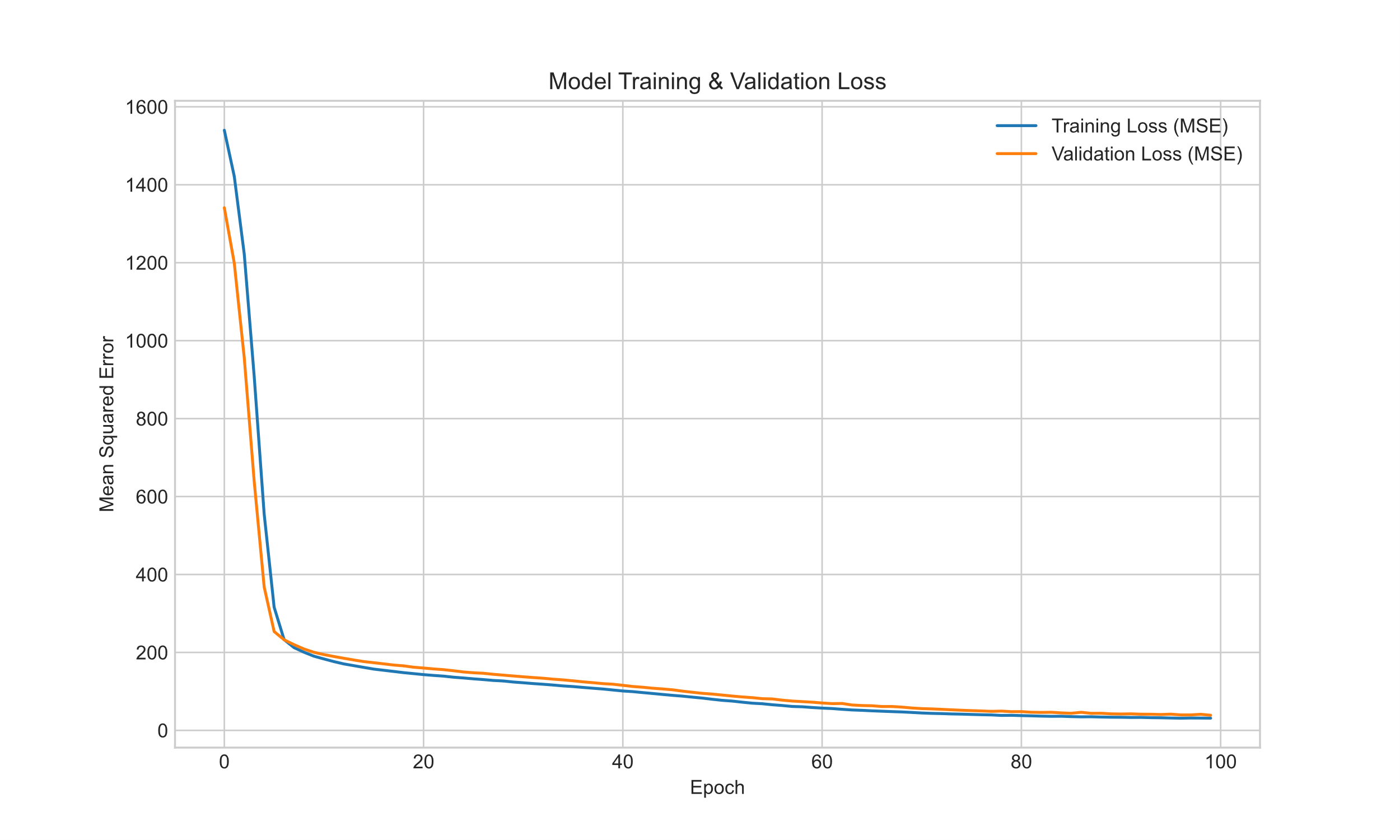
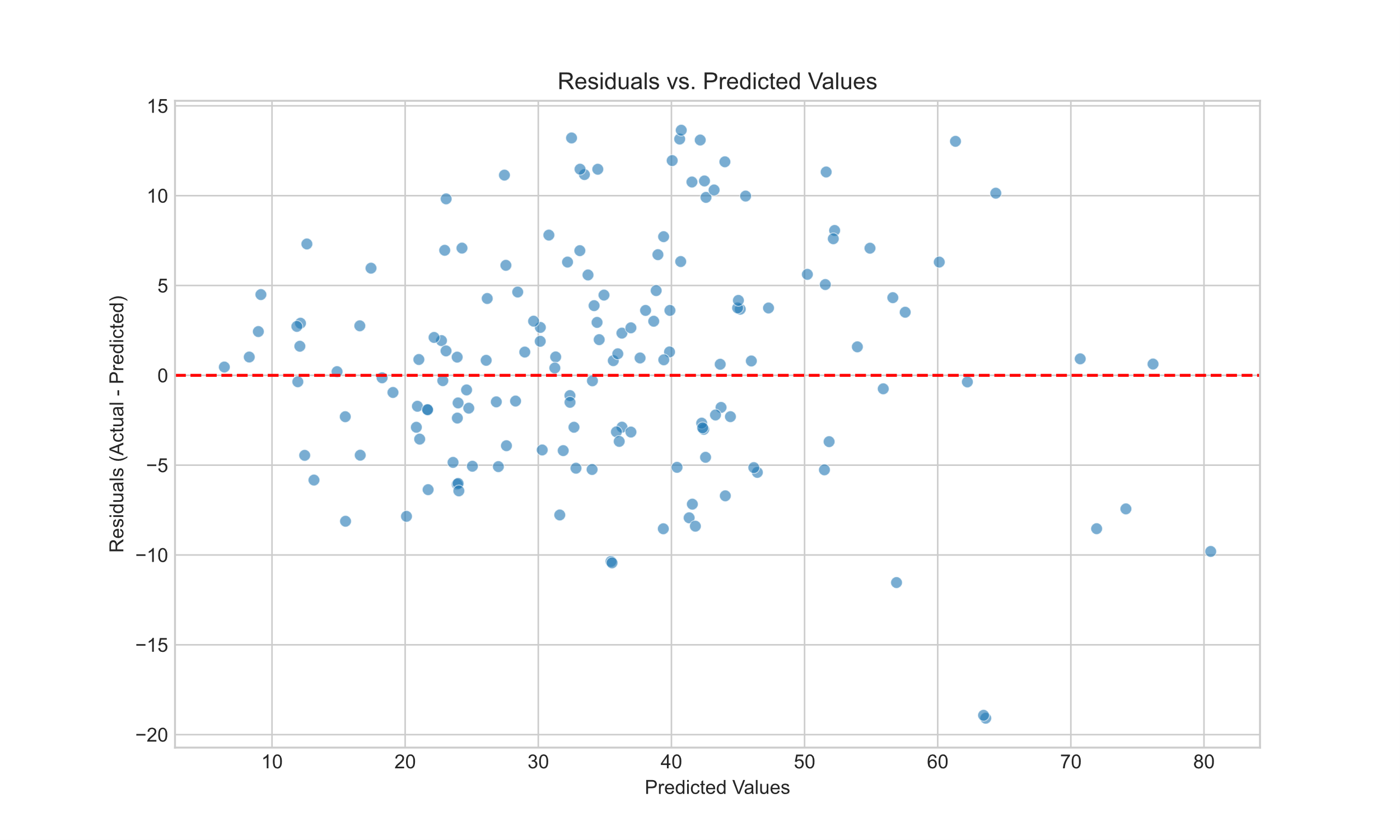
|  |  |
| --- | --- |
| Metric | Value |
| Mean Squared Error (MSE) | **39.91** |
| Mean Absolute Error (MAE) | **5.01** |
| R-squared (R2) | **0.841** |
| Mean Absolute Percentage Error (MAPE) | **15.75%** |

Export to Sheets

### 4.2. Analysis of Results

The results indicate a good level of performance for a simplified model.

* An **MAE of 5.01 MPa** signifies that, on average, the model's prediction of compressive strength is off by approximately 5.01 MPa. While slightly higher than the previous model, this level of error is still generally considered **good** in the context of civil engineering, suggesting a reliable predictive tool.
* The **R-squared (R2) score of 0.841** is strong. It implies that approximately **84.1%** of the variability in the actual concrete compressive strength is explained by the model. This high value confirms a **strong correlation** between the model's predictions and the actual values, indicating a robust and well-fitting model, especially given its architectural simplicity compared to the original nine-hidden-layer design.
* The **MAPE of 15.75%** provides a relative measure of error, showing that the average prediction error is about 15.75% of the actual value, which is a respectable figure.

****

## 5. Conclusion

### 5.1. Summary of Work

This project involved the design, training, and evaluation of a deep neural network to predict concrete compressive strength based on its constituent ingredients and age. A **simplified three-layer network** (two hidden layers with 64 neurons each) was developed and trained using the Adam optimizer with a learning rate of **0.001**. The model's performance was rigorously assessed on a held-out test set, yielding strong results across multiple standard regression metrics.

### 5.2. Final Conclusion

This project successfully demonstrated that an artificial neural network, even with a relatively **shallow architecture**, can be an effective tool for predicting concrete compressive strength with a high degree of accuracy. The final model achieved an R-squared score of **0.841** and a Mean Absolute Error of **5.01 MPa**. This performance confirms a robust and well-fitting model, suggesting its potential as a valuable asset for material science and civil engineering applications, enabling faster and more cost-effective formulation analysis.

### 5.3. Future Work

While the current model is effective, several avenues exist for potential improvement:

* **Experiment with Other Architectures**: Although the simplified network performed well, explore other deep learning designs, such as **Convolutional Neural Networks (CNNs)** for tabular data or deeper fully connected networks, to see if they can capture more complex non-linearities.
* **Investigate Other Models**: Investigate the performance of other machine learning algorithms, such as **Gradient Boosting (e.g., XGBoost, LightGBM)** or **Random Forests**, which are often strong performers on tabular data and may outperform the current neural network.
* **Hyperparameter Tuning**: Conduct a more extensive hyperparameter search, particularly for the **learning rate (0.001)** and the **number of neurons (64)** in the hidden layers, using techniques like Grid Search or Bayesian Optimization to find an even more optimal configuration.
* **Deployment**: Develop a simple web application or API that allows users to input concrete mixture details and receive an instant prediction of its compressive strength, making the model accessible to engineers and researchers.

### .

Data set site: https://www.kaggle.com/datasets/maajdl/yeh-concret-data?resource=download