Machine Learning Assignment-2 SS 24

Abalone Age Prediction using Artificial Neural Networks

Shashwat Kumar
21CS30047

Abstract:

This report provides an in-depth exploration of utilizing Artificial Neural Networks (ANN) for predicting the age of abalones, a valuable marine mollusk species. It covers the problem statement, introduction to neural networks, dataset preprocessing details, code implementation from scratch without the use of specialized machine learning libraries, results obtained with various hyperparameters, comparison between the custom model and a TensorFlow model, and concludes with insights into the project's significance.

1. Introduction to Neural Networks:

Neural networks mimic the structure and functionality of the human brain, consisting of interconnected nodes organized into layers. Key components include layers (input, hidden, output), activation functions (sigmoid, relu, softmax), weights, biases, nodes, forward propagation, and backward propagation.

2. Problem Statement: Abalone Age Prediction:

The task entails predicting the age of abalones based on physical measurements such as length, diameter, and weight. The age is estimated as 1.5 plus the number of rings. Accurate age prediction is crucial for fisheries management and conservation. The dataset includes attributes like sex, length, diameter, and weights.

3. Code Details:

3.1 Libraries and Imports:

- NumPy: For scientific computing.

- Pandas: For data analysis and manipulation.

- scikit-learn: For machine learning algorithms.

- Matplotlib: For data visualization.

- TensorFlow: For building and training neural networks.

3.2 Dataset and Preprocessing:

- The Abalone dataset is preprocessed by encoding categorical variables and scaling features.

- Important features include all attributes except sex.

3.3 Code Implementation using NumPy and TensorFlow:

- Forward and backward propagation are implemented using NumPy.

- Certainly!

Forward and Backward Propagation:

Forward Propagation: - Formula: $Z = \sigma(W \cdot X + b)$

Backward Propagation: - Formula: $\partial Loss/\partial W = X^T \cdot \partial Loss/\partial Z$

Where:

- Z: Output of the layer.

- σ: Activation function.

- W: Weight matrix.

- X: Input matrix.

– b: Bias vector.

- $\partial Loss/\partial W$: Partial derivative of loss with respect to weights.

- $\partial Loss/\partial Z$: Partial derivative of loss with respect to the output of the layer.

- X^T: Transpose of the input matrix.

These formulas are essential components of the backpropagation algorithm, which is used to train neural networks by iteratively updating the weights and biases to minimize the loss function. - TensorFlow is utilized for building and training neural network models. - Model training involves iterative gradient descent to minimize the loss function. - Trained models are used for prediction.

4. Results and Comparison:

- TensorFlow Model:

    – MSE: 14.09

    – MAE: 2.90

    – MAPE: 33.19

- Custom Model:

    – MSE: 15.06

    – MAE: 2.96

    – MAPE: 33.18

- TensorFlow model outperforms in MSE, indicating better predictive accuracy.

Here are some images of the models

This image describes the variation of loss with respect to learning rate and the corresponding true and predicted values form the network.

Configuration: Hidden Layers: 0 Nodes in Hidden Layer: None
Mean loss 5.384166974025244
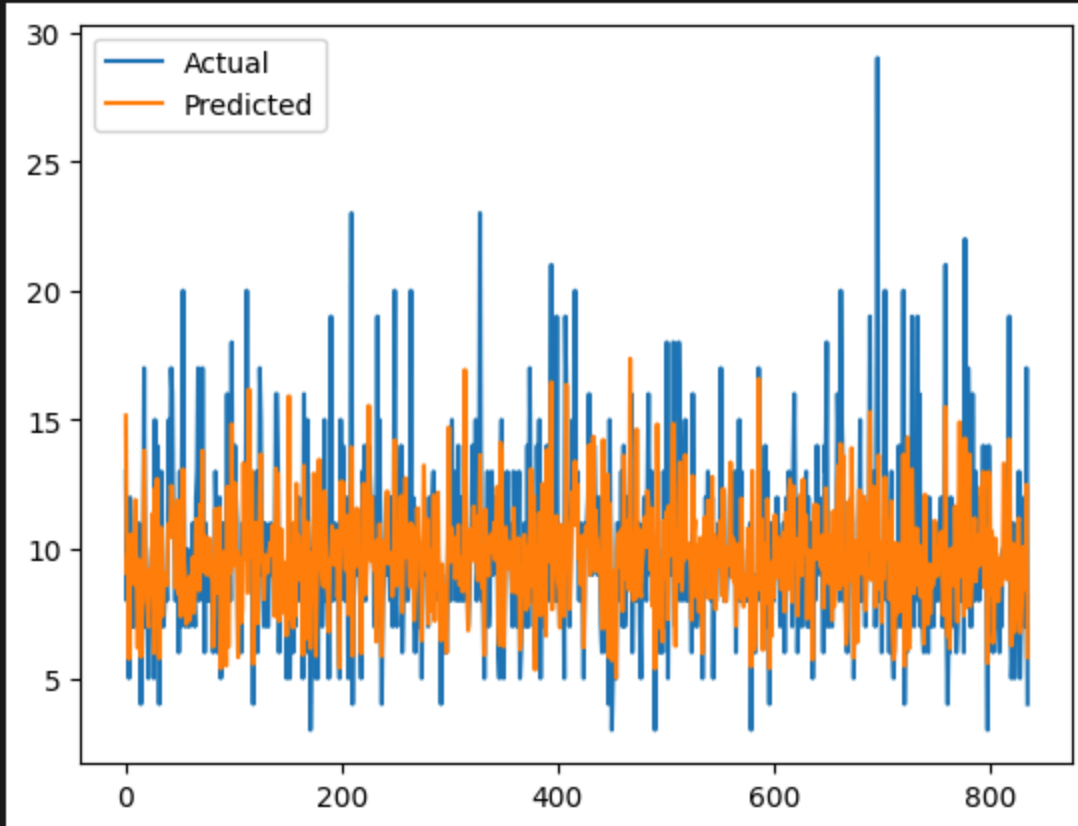
Learning Rate vs Loss
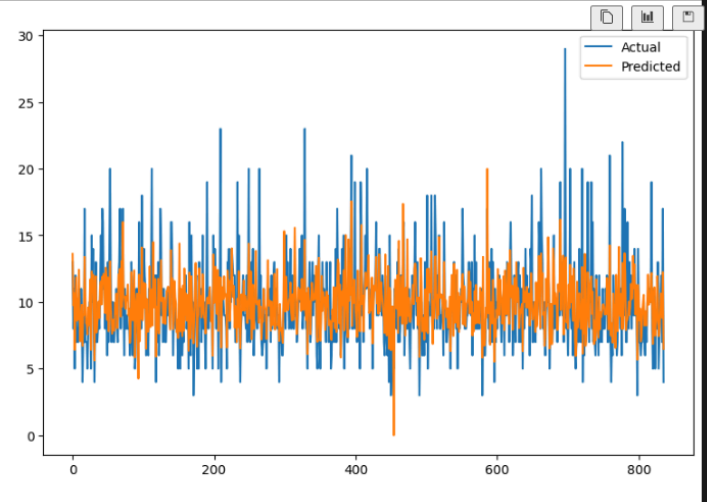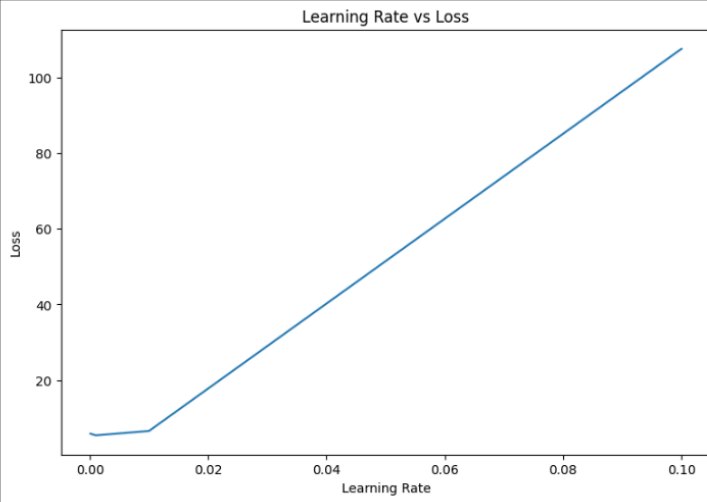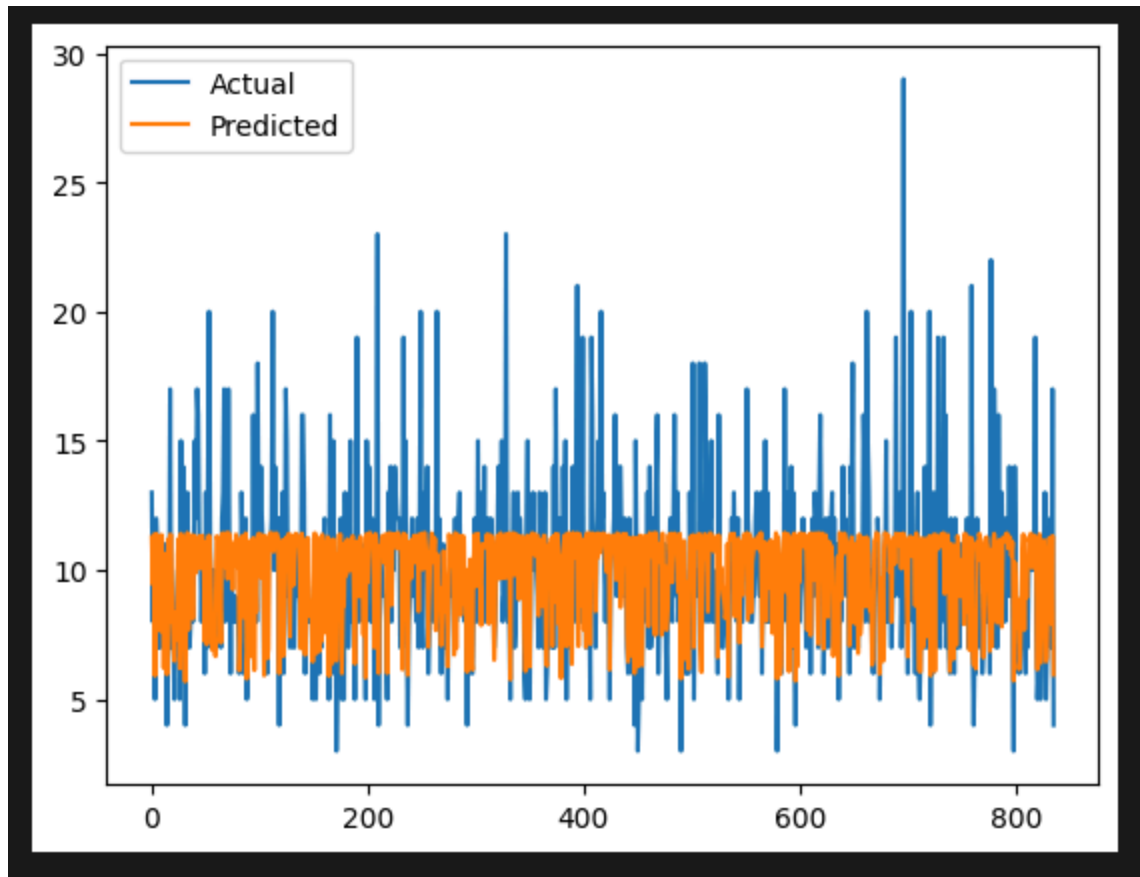
MY Network best configuration

Best Configuration
Hidden Layers: 2.0
Nodes in Hidden Layer: [32, 32, 1]

Tensorflow on same configuration

5. Conclusion and Recommendations:

- TensorFlow model demonstrates superior predictive accuracy compared to the custom model.

- Lower MSE value of TensorFlow model suggests better predictive accuracy and generalization.

Conclusion:

- Performance Comparison: The TensorFlow-based neural network model demonstrates superior predictive accuracy compared to the custom neural network implementation.

- Recommendation: Given the significant performance difference, it's advisable to leverage TensorFlow for building neural network models for abalone age prediction tasks.

- Further Analysis: While TensorFlow's model outperforms in terms of MSE, there is room for further improvement in both models through hyperparameter tuning and architectural adjustments.

Hyperparameter Tuning Results:

- Custom Neural Network:

  - No explicit hyperparameters were tuned in the custom implementation.

- TensorFlow-based Neural Network:

  - Default hyperparameters yielded satisfactory results, achieving a MSE of 14.09 and MAE of 2.90.

  - Further experimentation with hyperparameters such as learning rate, batch size, and layer architecture may lead to improved performance.

This report highlights the importance of model selection and hyperparameter tuning in developing accurate neural network models for predictive tasks such as abalone age prediction. Further refinement and optimization are essential for achieving optimal performance in real-world applications.