Regression and Classification of Abalone Age

*Comparison of three different models(Linear Regression, Logistic Regression and Neural Network)

1st Yuting Fang
MATH3856
University of New South Wales
Sydney, Australia
z5518340@ad.unsw.edu.au

1st Bingchen Xie
MATH5836
University of New South Wales
Sydney, Australia
z5610033@ad.unsw.edu.au

1st Runyu Han
MATH3856
University of New South Wales
Sydney, Australia
z5387147@ad.unsw.edu.au

***Abstract*—This paper presents a solution to regression and classification problems on an abalone physical characteristics data set. The objective was to predict the age of abalone with certain characteristics and to classify whether the age is greater than or equal to 7 or less than 7. The performance of linear regression, logistic regression, and neural networks in solving regression or classification problems was compared, and the use of hyper-parameters of neural networks was tested as best in our experiments. Ultimately, the findings reveal that linear regression is most effective for regression, while logistic regression remains the preferred choice for classification. Furthermore, after data normalization, the results demonstrated notable improvements, with the lowest RMSE (2.168725) achieved in linear regression with the normalized all features on test set, while the highest accuracy of 93.6% was attained in age classification with logistic regression with the normalized all features.**

***Index Terms*—Linear Regression, Logistic Regression, Neural Network, Abalone, Normalization, hyper-parameters**

I. Introduction

A. Background

A neural network enables a machine to learn data by imitating the processes of human learning [1]. It is precisely due to the impressive capabilities of this technology that it has gained widespread recognition [2]. It has applications in a very wide range of fields, from data-based predictions to face, voice, and image recognition, which undoubtedly have a significant impact on our daily lives [3]. Additionally, a significant challenge in the design of neural network (NN) systems is the determination of the optimal structure and hyper-parameters for the network, given the data pertinent to the machine learning problem at hand [4].

Furthermore, as the world's population grows, the global market for abalone, one of the most famous and sought-after seafood delicacies, has grown significantly in recent years and the abalone industry has become an important part of the global economy [5]. The age of the abalone is highly correlated to its price, as it is the only factor used to determine its value [6]. Technically, the rings are the result of the gradual growth of the abalone's inner shell at a rate of one ring per year [6].

In light of this challenge, a few scientists have endeavored to employ neural networks as a means of addressing the issue of abalone age prediction and classification. A review of the literature reveals that the artificial intelligence approaches currently available for abalone age prediction are still inadequate [6].

B. Purpose

Based on the above background, we can make a comparison between the neural network and the preceding model and explore the methods of finding the best hyper-parameters on the abalone data set, to determine whether a notable enhancement compared to linear models in regression and classification has been achieved. This approach allows us to address the issue of age prediction in abalone while simultaneously evaluating the efficacy of diverse methodologies. It enables us to identify the optimal prediction method and achieve superior prediction outcomes.

C. Contribution

To determine the age of the abalone efficiently and also compare the performance of the neural network with some linear models, this article will discuss the different regression and classification methods, which are used to predict or classify the abalone's age based on eight physical features or two chosen features, respectively. Furthermore, the article will present a methodology for selecting the optimal regression and classification method among the three given models. The details are as follows.

- The original data is a biology data set about Abalone from UC Irvine, used for predicting the ring age of abalone. Inside this data, there are 4177 instances of eight features "Sex", "Length", "Diameter", "Hight", "Whole weight", "Shucked weight", "Viscera weight", "Shell weight" of abalone, and the "Rings" to show the ring age of abalone. All these variables have the continuous type, except the "Sex" data which are categorical to "M(Male)", "F(Female)" and "I(Infant)" and the "Rings" data are integers.
- Trying to find the best hyperparameter of neural network for both regression and classification. After that, for the regression part, we compare the linear regression and the neural network and consider three different scenarios, such as using all original features in the data set, using all features with normalization, and using two features that

are most relevant to 'ring' respectively. For the classification part, we compare the logistic regression and neural network in the same three scenarios with regression. The objective is to identify the optimal methodology for addressing classification and regression problems and the impact of normalization. .

II. Methodology

A. Data Processing

- Data Cleaning: As the original dataset contains no missing values, we only process the data by altering the categorical feature "Sex" to be three one-zero features "M(Male)", "F(Female)" and "I(Infant)" to support continuous process. For instance, if an abalone is male, in the data, the "M" would be '1', but the "F(Female)" and "I(Infant)" would be '0'.
- Correlation: The figure 1 explores the correlation between the "Rings" with other features and reveals that the "Rings" have the highest positive correlation value with features "Diameter" (0.57) and "Shell weight" (0.63). Generally, selecting features with a high correlation to the response would improve the performance of linear model. However, according to Theresa Ullmann, selecting only a part of the features rather than all might lead to problems [7]. Therefore, we would carefully seek whether using both two features that have the highest correlation value with "Rings" in the regression is better than using all features.

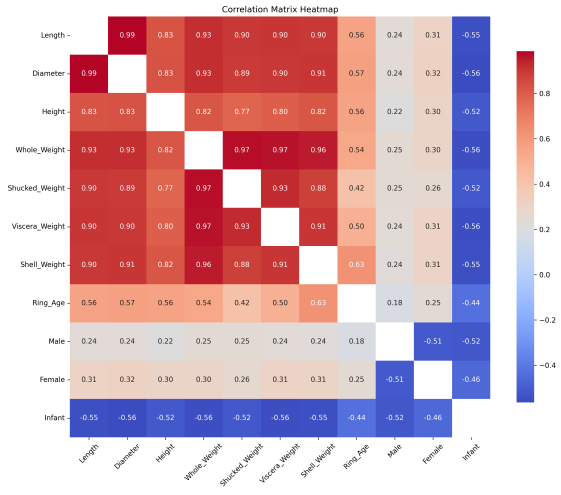
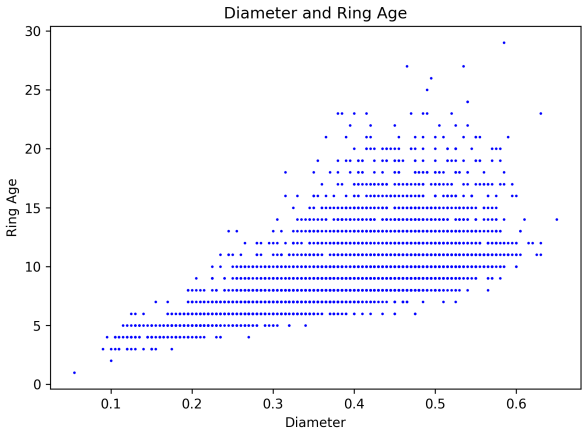
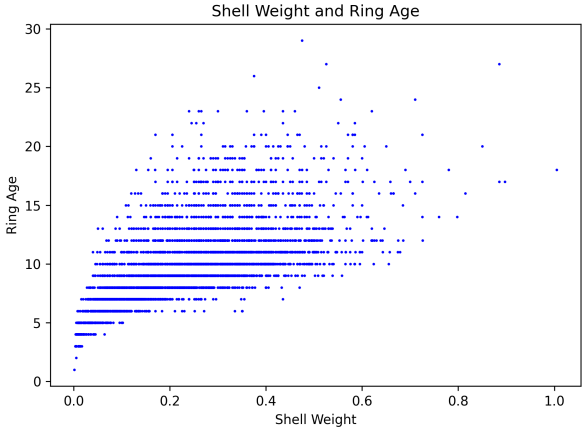


Fig. 1: Heatmap

- Scatter Plot: Figure 2 show the "Rings" are positively correlated with both "Shell weight" and "Diameter". However, the linear relationship between "Rings" and "Shell weight" or "Diameter" is not apparent. The trend of the scatter is not close to a line, but close to a fan-shape. Also, the fan is built with several straight lines perpendicular to the y-axis "Rings". That is because the abalones with the same "Rings" tend to have different "Shell weight" and "Diameter" in a wide range. Except that, there are no obvious outliers inside the scatter plot, but the abalones with high "Rings", "Shell weight" and "Diameter" have a sparser distribution than the lower ones. Especially the "Rings", most abalones have "Rings" below twenty.



(a) Diameter and ring plot

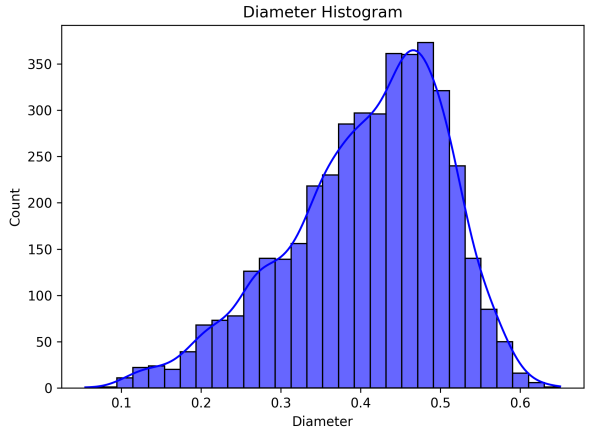


(b) Shell weight and ring plot

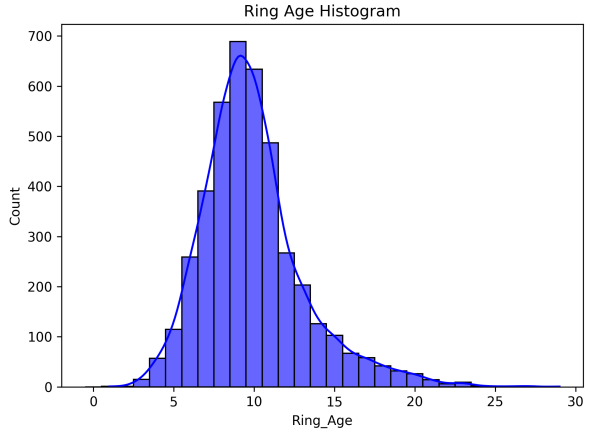
Fig. 2: Scatter plots of diameter, shell weight, and ring age

- Histogram: Figure 3 show all these three variables are not strict normal distributions. Figure 3b shows the distribution of "Rings" is close to normal distribution; however, it looks skewed to the left. The "Rings" only has one peak at 9, and a long tail at the right. Most data concentrate on the range from 5 to 20, and the data with "Rings" above 20 might be considered as the outliers. Figure 3c is like the "Rings Age Histogram". Most data are concentrated on the left, and there is also a long tail on the right, but the difference is that the "Shell Weight Histogram" has two peaks where one is close to 0.2 and one is close to 0.3. Figure 3a reveals that the graph skews to the right but has no long tail problem. Meanwhile, the "Rings Age Histogram" apparently has a higher density. "Rings Age Histogram" has around 650 counts concentrated on the

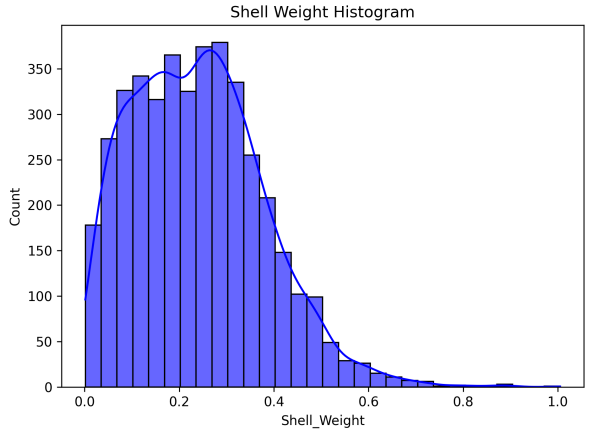
peak, which is much larger than around 350 counts of "Shell Weight Histogram" and "Diameter Histogram". Since histograms show these variables are not normal distributions, we consider not using the method that relies on the data's normal distributions when normalizing data.



(a) Diameter histogram



(b) Rings histogram



(c) Shell weight histogram

Fig. 3: Histograms for diameter, rings, and shell weight

B. Linear Model

- Linear regression is a method used to predict data, which relies on catching the linear relationship between variables, with the function

$$Y = X\beta + \epsilon \tag{1}$$

, where Y is the response vector, X is the feature matrix, β is the weight vector and ϵ is the error term. Training linear regression relies on minimizing the loss-function (Mean Squared Error)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - (mx_i + b))^2 \tag{2}$$

to find the best weight and biases.

- Logistic regression is generally used in binary classification problems. Using the sigmoid function

$$y = \frac{1}{1 + e^{-ux+b}} \tag{3}$$

to classify the input into two classes based on probability, larger or smaller than 0.5. For training logistic regression, We use Binary cross-entropy

$$L = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{4}$$

as the loss function.

- Gradient Descent: Gradient descent is the method used to find the best value for weight $\mathbf{w}$ and bias b to minimize the loss function in linear and logistic regression, requiring the following equation to update the weight W and bias b until the loss function is minimized.

$$\mathbf{w}_t := \mathbf{w}_{t-1} - \alpha\nabla_{\mathbf{w}} loss(\mathbf{w}, b) \tag{5}$$

$$b_t := b_{t-1} - \alpha\nabla_b loss(\mathbf{w}, b) \tag{6}$$

where α is the learning rate, t is the number of iterations and the gradient is the first derivative of each parameter in the loss function, which is

$$\nabla_{\mathbf{w}} loss(\mathbf{w}, b) = \frac{d}{d\mathbf{w}}\text{loss}(\mathbf{w}, b) \tag{7}$$

$$\nabla_b loss(\mathbf{w}, b) = \frac{d}{db}\text{loss}(\mathbf{w}, b). \tag{8}$$

C. Neural Network

Artificial neural network is one of the most common research methods in the field of artificial intelligence. We used a simple neural network, the multi-layer perceptron, consisting of an input layer, an output layer and a hidden layer. The input data is propagated forward through the hidden layers to the output layer, which takes these internally generated features and turns them into the final output result.

The backpropagation algorithm is one of the fundamental methods for training neural networks. It is based on gradient descent, but backpropagation adjusts for each layer by calculating the gradient of the loss function for each weight and

bias through the chain rule [8]. The operational procedures are as follows:

- Calculate the error at the output layer. $\delta^{(l)}$ is the error term of the output layer, $f'\left(z^{(l)}\right)$ is the derivative of the activation function at the l layer.

$$\delta^{(L)} = \frac{\partial L}{\partial a^{(L)}} \cdot f'\left(z^{(l)}\right) \tag{9}$$

- The error is then backpropagated to the previous layer, calculating the error for each layer in turn. W is the weight vector, $\left(W^{(l+1)}\right)^T \delta^{(l+1)}$ is the backpropagated error from the $(l+1)$ layer.

$$\delta^{(l)} = \left(W^{(l+1)}\right)^T \delta^{(l+1)} \cdot f'\left(z^{(l)}\right) \tag{10}$$

- Calculate weights and bias gradients. $a^{(l-1)}$ is the transpose of the activation from the $(l-1)$ layer

$$\frac{\partial L}{\partial W^{(l)}} = \delta^{(l)} \cdot \left(a^{(l-1)}\right)^T \tag{11}$$

$$\frac{\partial L}{\partial b^{(l)}} = \delta^{(l)} \tag{12}$$

- Using Stochastic Gradient Descent to update weights and bias:

$$W^{(l)} = W^{(l)} - \eta \frac{\partial L}{\partial W^{(l)}} \tag{13}$$

$$b^{(l)} = b^{(l)} - \eta \frac{\partial L}{\partial b^{(l)}} \tag{14}$$

Stochastic Gradient Descent (SGD) is a method that randomly extracts a sample from the training set to calculate the gradient of the loss function, which can greatly improve the running speed:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta L(\theta; x^{(i)}, y^{(i)}) \tag{15}$$

The path of stochastic gradient descent is unstable, so we use momentum and Mini-Batch SGD to optimize it. The momentum term can make the parameter update take into account not only the current gradient but also the previous accumulation, which can reduce the oscillation and accelerate the convergence [9]. Mini-Batch SGD is a compromise between SGD and full gradient descent. By dividing the dataset into smaller batches, each small batch contains m samples, and selecting one of the small batch data to update the parameters during each gradient descent.

$$v_t = \beta v_{t-1} + (1-\beta)\nabla_\theta L(\theta; x^{(i)}, y^{(i)}) \tag{16}$$

$$\theta_t = \theta_{t-1} - \eta v_t \tag{17}$$

- t is the iteration number,
- v_t is the velocity or momentum at time step t,
- β is the momentum coefficient,
- η is the learning rate,
- $\nabla_\theta L(\theta; x^{(i)}, y^{(i)})$ is the gradient of the loss function with respect to the parameter θ.

D. Normalization

Normalization is a data preprocessing technique that adjusts the scales of different features to similar or normalized scales. And the normalization method we used in the paper is Min-Max Normalization and Vector Normalization (also known as Normalizer).

- Min-Max Normalization:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{18}$$

  where x' is the element for a feature, $x_{\max}$ is the maximum value in the feature and $x_{\min}$ is the minimum value in the feature, and all the elements in the feature. process by the formula.
- Vector Normalization (Normalizer):

$$x' = \frac{x}{\|x\|} \tag{19}$$

  x is the vector of the sample in the data set and $\|x\|$ is the L2-norm of it.

Based on program trials, we compared the effects of different normalization methods and selected the Vector Normalization for regression tasks and min-max normalization for classification tasks.

E. Software Suite

We used some python modules and packages that are common in data analysis and machine learning, including os, time, numpy, pandas, sklearn, seaborn, matplotlib, itertools and tensorflow.

More specifically, sklearn helped us with linear model-related tasks, as well as tensorflow helps us with neural network-related tasks. In sklearn, the preprocessing package can normalize the dataset, the model_selection package can split the dataset into training and test parts, linear_model package can fit linear models, and the metrics package can compute various performance metrics. In tensorflow, we primarily utilized the keras package, which is modular and easy to operate.

F. Experiment Setting

In terms of classification tasks, we established a rule that classify samples into two classes. One is larger and equal to 7, as well as another is less than 7.

For regression and classification, we built linear models and neural networks, respectively, to fit following three forms of datasets. The first one was not normalized and used all features. The second one was normalized and used all features. The third one was not normalized and used only diameter and shell weight as input features. Overall, we built 12 models and compared their performance. All model settings and dataset settings are presented in Table I.

For each model, we took 60% of the dataset as the training set and the remaining 40% as the test set. This process was repeated 30 times with different random seeds obtain various data splits. After fitting a model, we used it to predict outcomes

for both the training set and test set. We calculated the Root Mean Square Error (RMSE) and R-squared for the regression task, as well as Accuracy, AUC score, Area Under the Receiver Operating Characteristic Curve (AUC) and PR-curve for the classification task. Ultimately, we computed the mean and standard deviation of the aforementioned metrics.

When finding the best hyper-parameters for neural networks, we design the following trial experiments, repeating 4 times for different settings (without considering normalization) of neural networks. (The number of units in the input layer depends on the number of input features, in the output layer we use sigmoid function for classification and Linear function for the regression and in we setting the epochs as 200).

- Two layers: for the activate function on the input layer we test 'relu'

$$\text{ReLU}(x) = \max(0, x) \tag{20}$$

  and 'sigmoid' separately with a learning rate from 0.1 to 0.001, like 0.1, 0.09, 0.08,.....
- Three layers: for the activate function on the input and one hidden layer using the same testing method as above, but using a different number of units for the hidden layer, trying to gradually increasing the number and gradually decreasing the number respectively
- Four layers: using the same testing method above.
- Five layers: also using the same testing method above.

As for the loss function, we use MSE for regression tasks and Binary Cross-Entropy Loss for classification tasks, which are already mentioned above. Furthermore, we use the stochastic gradient descent, with 0.9 momentum, and divide the training set into 32 mini-batches to accelerate the convergence of the minimizer, when minimizing the loss function.

TABLE I: Model and Dataset Settings

Model Type	Task Type	Normalization	Input Features
Linear Model	**Regression**	**None**	**All Features**
Linear Model	**Regression**	**Normalizer**	**All Features**
Linear Model	**Regression**	**None**	**Two Features**
Linear Model	**Classification**	**None**	**All Features**
Linear Model	**Classification**	**MinMaxScaler**	**All Features**
Linear Model	**Classification**	**None**	**Two Features**
Neural Network	**Regression**	**None**	**All Features**
Neural Network	**Regression**	**Normalizer**	**All Features**
Neural Network	**Regression**	**None**	**Two Features**
Neural Network	**Classification**	**None**	**All Features**
Neural Network	**Classification**	**MinMaxScaler**	**All Features**
Neural Network	**Classification**	**None**	**Two Features**

III. Result

This section presents the findings of the three models, which are dependent on the methodology employed. The experimental results demonstrate that all methods exhibited comparable performance on both the test set and the training set. Consequently, it can be stated that those models demonstrated the desired degree of generalization. Furthermore, it offers a comparative analysis of the models' respective advantages and disadvantages in the end.

A. Best Hyper-parameters

In the trial experiment for neural networks, it was observed that when training three-layer and four-layer neural networks for the same training set, the accuracy remained consistent, as did the precision and recall curves. Additionally, it was determined that an increase in the number of layers and units would result in over-fitting, which suggests that the model exhibits a lack of generalization ability, performing significantly better on the training set than on the test set. Accordingly, in light of the considerable computational burden and time expenditure involved, we ultimately opted to utilize a three-layer neural network. Following an extended period of rigorous debugging, we were able to ascertain the units and activation functions associated with each layer and the learning rate for all scenarios. Above all are all shown in Table II In neural networks dealing with regression tasks after several

TABLE II: Details of Neural Network

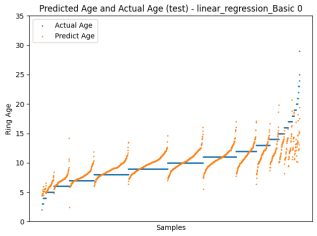
Type	Number of unit in each layer			
	layer1	layer2	layer3	Learning Rate
All Feature R	10	8	1	0.001
All Feature C	10	12	1	0.09
Two feature R	2	4	1	0.001
Two Feature C	2	4	1	0.09

trial experiments, we use 'relu' (formula 20) as the activation function of both the input and hidden layer, and in the output layer we use the linear activate function. In neural networks dealing with classification tasks, we use 'relu' as the activation function of the hidden layer, and the sigmoid function as the activation function of the hidden and output layer. In the end, we set all the epochs as 100 and used the early-stopping code to stop the upgrading when a superior outcome can not be achieved within the following 5 epochs.

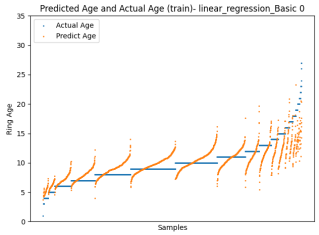
B. Regression

In this section, we used linear models and neural networks across three scenarios. The basic scenario is using all input features to train the model. Another one is using all normalized input features. The last one is using only the two most relevant input features: diameter and shell weight. The results of the regression tasks are illustrated in Figures 4, 5, and 6. The model's performance on the training set and the test set is very close, which shows that the model has good generalization ability.

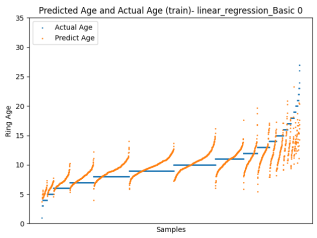
- When making age regression predictions, these models exhibited some common trends: for samples with an actual age that is too low, the predicted age is generally higher than the actual age, while for samples with an actual age that is too high, the predicted age is typically lower than the actual age. In contrast, these models tend to produce predictions that are closer to the actual values for samples whose actual ages are nearer to the mean age of the dataset, known as Regression to the Mean (RTM) [10].
- According to Tables III, we observe that when using all features, neural networks have a slightly better perfor-



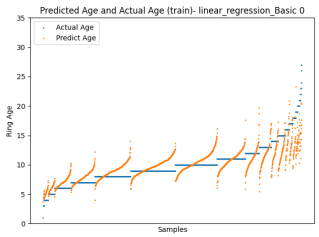
(a) LR(test)



(b) LR(train)

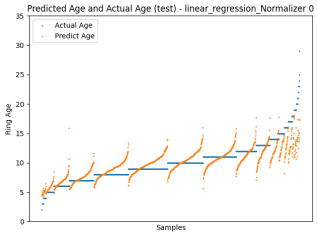


(c) NN(test)

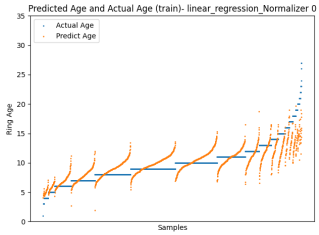


(d) NN(Train)

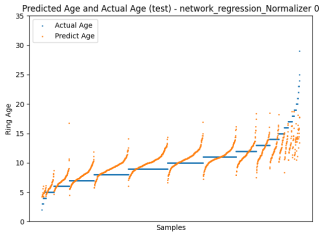
Fig. 4: Predict/Actual Age (All Features)



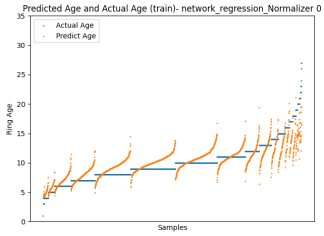
(a) LR(test)



(b) LR(train)



(c) NN(test)

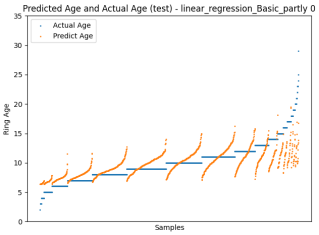


(d) NN(Train)

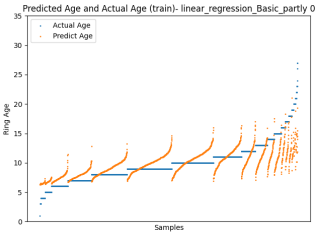
Fig. 5: Predict/Actual Age (All Normalized Features)

mance with smaller RMSE and larger R-squared. For example, the linear model using all features has a better performance on the training set with 0.542 R-squared value than the performance on the test set with 0.523 R-squared value.

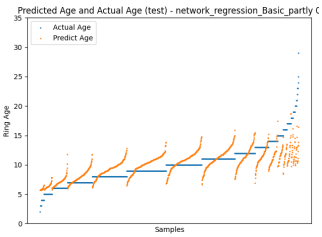
- Figure 5 and 4 reveals when using normalization, the predicted ages of older samples are more concentrated around lower age values. The higher predicted values in the regression without normalization, which are closer to the actual ages, are significantly diminished in the case with normalization. Besides, figure IV vector normal-



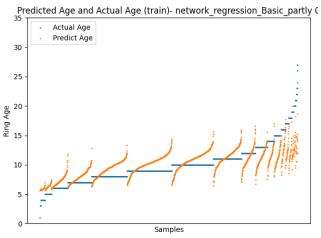
(a) LR(test)



(b) LR(train)



(c) NN(test)



(d) NN(Train)

Fig. 6: Predict/Actual Age (Two Features)

TABLE III: The mean and std values for RMSE and R-squared for 30 experiments (All features, without Normalization)

Method	**RMSE**		R^2	
	Mean RMSE	**Std RMSE**	**Mean** R^2	**Std** R^2
LR (Test)	2.228689	0.050434	0.523105	0.021737
LR (Train)	2.178460	0.032143	0.542020	0.011063
NN (Test)	2.168466	0.055023	0.548660	0.019022
NN (Train)	2.144925	0.059353	0.555869	0.021482

ization significantly improves the performance of linear models but harms neural networks, making the linear regression (RMSE: 2.168725, R-squared: 0.548555) outperform the neural network (RMSE: 2.182012, R-squared: 0.539359) on both test and training sets.

- Table V and 6 shows that as same settings, using only two input features decreases the performance, with a notably increased RMSE on the test set for neural network, from

TABLE IV: The mean and std values for RMSE and R-squared for 30 experiments (All features, with normalization)

Method	**RMSE**		R^2	
	Mean RMSE	**Std RMSE**	**Mean** R^2	**Std** R^2
LR (Test)	2.168725	0.046867	0.548555	0.016920
LR (Train)	2.122621	0.031307	0.565213	0.009731
NN (Test)	2.182012	0.196453	0.539359	0.101428
NN (Train)	2.164635	0.203401	0.544210	0.101786

TABLE V: The mean and std values for RMSE and R-squared for 30 experiments (Two features)

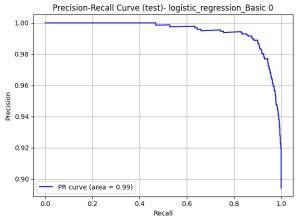
Method	**RMSE**		R^2	
	Mean RMSE	**Std RMSE**	**Mean** R^2	**Std** R^2
LR (Test)	2.521342	0.059516	0.390008	0.019062
LR (Train)	2.502071	0.039277	0.395904	0.012726
NN (Test)	2.706703	0.313840	0.287221	0.174146
NN (Train)	2.690265	0.326498	0.291971	0.176753

2.168 to 2.707. In neural networks, where a large number of parameters need to be optimized, more comprehensive information can benefit a lot. Reducing the number of input features makes the model get insufficient information, resulting in under-fitting [11].

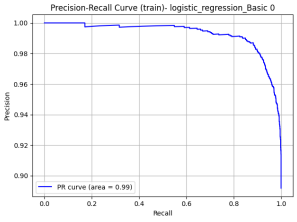
From the current findings, a neural network performs slightly better than linear regression when the given information is enough and adequately used. However, in all scenarios, the best performance of linear regression and neural networks are roughly the same, and considering all the conditions, the neural network always costs more time, so we decide to use linear regression with normalization.

C. Classification

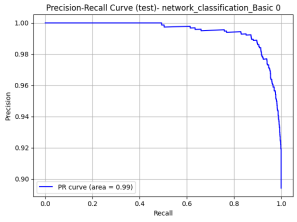
In this part, we classify the age of abalone into larger and equal to seven and less than seven and compare the logistic regression and neural networks, under three similar scenarios in regression works.



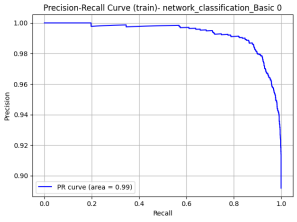
(a) LRC(test)



(b) LRC(train)



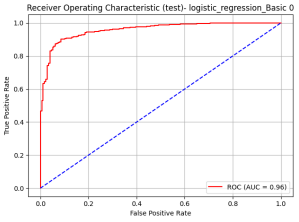
(c) NN(test)



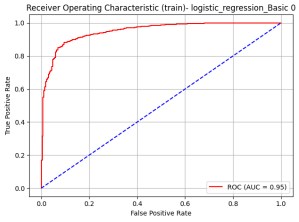
(d) NN(train)

Fig. 7: PR Curve for all features classification

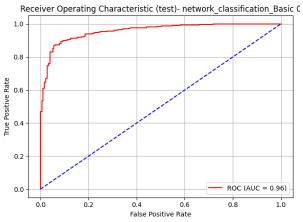
- In tableVI, we compared the performance of logistic regression and neural regression using all the features. It is observable that the LRC performs slightly better than NN on both testing and training sets, with higher mean accuracy and mean AUC. Simultaneously, the stabilization of accuracy is worse than LRC. Additionally, the figure 7 demonstrates that they roughly have the same PR-curves, and their under-line areas are all 0.99, which also reveals that LRC just has a slight advantage. Similarly, in the figure 8 two of the ROC curves show the same areas in the testing set, with 0.96 under-line area.
- From the tableVII, we can find normalization does improve the performance of LRC and NN on the mean accuracy(around 0.2%) and also the AUC score. Although the improvements are not clear on the figure 9 and figure10 which showing the PR-curve and the ROC-curve



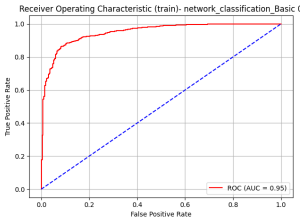
(a) LRC(test)



(b) LRC(train)



(c) NN(test)

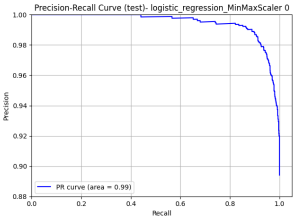


(d) NN(train)

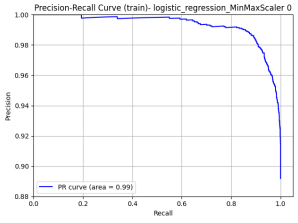
Fig. 8: ROC Curve for all features classification

TABLE VI: The mean and std values for accuracy and AUC for 30 experiments (all features)

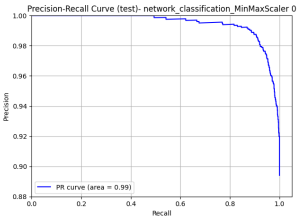
Method	Accuracy		AUC	
	Mean-Acc	Std-Acc	Mean AUC	Std AUC
LRC (Test)	0.934889	0.003898	0.952810	0.005454
LRC (Train)	0.934743	0.002678	0.953164	0.003263
NN (Test)	0.934171	0.005377	0.952343	0.005377
NN (Train)	0.934330	0.002903	0.952748	0.003418



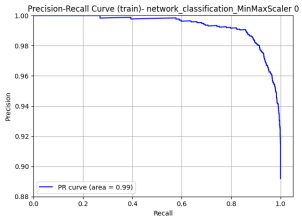
(a) LRC(test)



(b) LRC(train)



(c) NN(test)

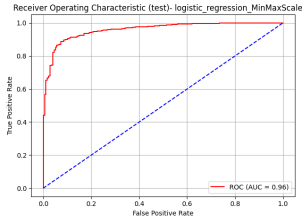


(d) NN(train)

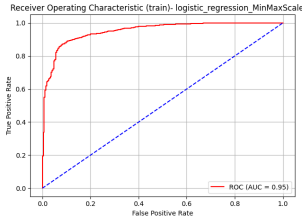
Fig. 9: PR Curve for all features classification with normalization

of them, and illustrating almost the same answer with Non-Normalied part. Furthermore, there are no notable different between LRC and NN.

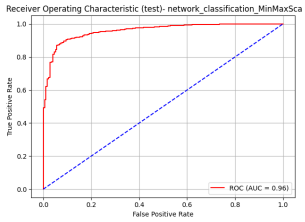
- Using the features "Diameter" and "Shell Weight" does not reveal any advantage, we observe from the table VIII



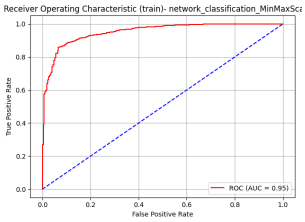
(a) LRC(test)



(b) LRC(train)

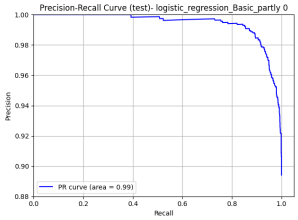


(c) NN(test)

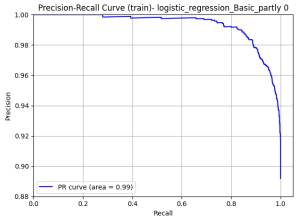


(d) NN(train)

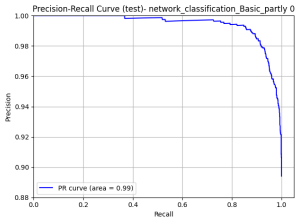
Fig. 10: ROC Curve for all features classification with Normalization



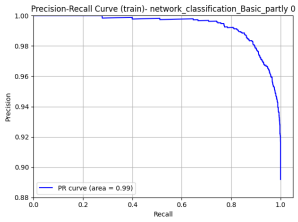
(a) LRC(test)



(b) LRC(train)



(c) NN(test)



(d) NN(train)

Fig. 11: PR Curve for two features classification

TABLE VII: The mean and std values for accuracy and AUC for 30 experiments (all features with normalization)

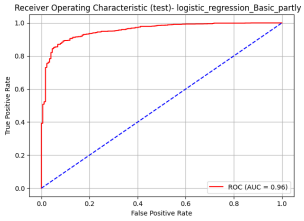
Method	Accuracy		AUC	
	Mean-Acc	Std-Acc	Mean AUC	Std AUC
LRC (Test)	0.936146	0.004463	0.954192	0.005295
LRC (Training)	0.935501	0.003052	0.954534	0.003185
NN (Test)	0.936326	0.003956	0.954140	0.005195
NN (Training)	0.936193	0.003144	0.954461	0.003349

that there are decrease of around 0.4% on mean accuracy for both LRC and NN. Furthermore, the PR-curve and ROC-curve in the table 11 and table12 also have roughly no differences between the above two.

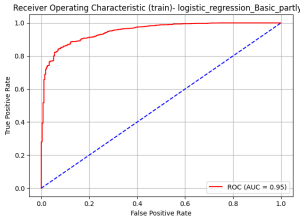
TABLE VIII: The mean and std values for accuracy and AUC for 30 experiments (two features)

Method	Accuracy		AUC	
	Mean-Acc	Std-Acc	Mean AUC	Std AUC
LRC (Test)	0.932755	0.005468	0.952973	0.004736
LRC (Training)	0.931205	0.002754	0.952325	0.003072
NN (Training)	0.930700	0.014139	0.892891	0.154164
NN (Test)	0.927547	0.015009	0.892110	0.153823

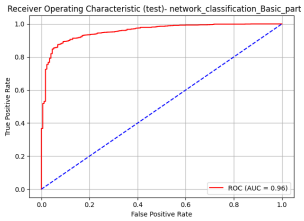
The results of the classification demonstrate that the accuracy and other evaluating methods consistently show no clear gap between LRC and NN. And as normalization indeed improves the performance of both of them. Concurrently, considering the cost of time, it suggests that LRC may be a more suitable approach than traditional neural networks for classification problems on this dataset, as neural networks are always slower than logistic regression in our experiments. Finally, we print the confusion matrix of LRC using all features with normalization in Figure 13, which also illustrates that the model does not suffer from over-fitting and maintains consistent generalization to unseen data.



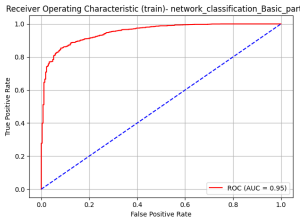
(a) LRC(test)



(b) LRC(train)



(c) NN(test)



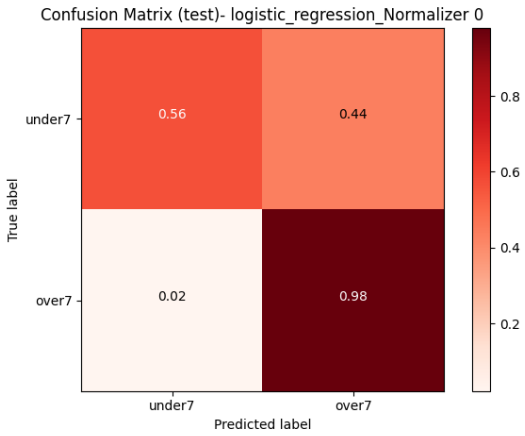
(d) NN(train)

Fig. 12: PR Curve for two features classification

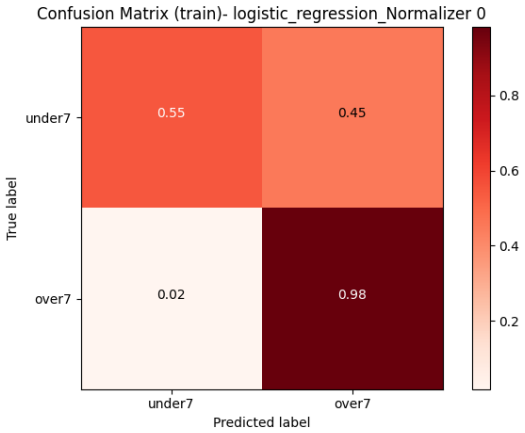
IV. Discussion

A. Summary

The objective was successfully attained, with satisfactory responses obtained in both the prediction of abalone age and the classification of specimens above and below the age of seven. Concurrently, the optimal methodology was identified among the three selected models and input features, exhibiting the (RMSE) of 2.168725 from linear regression with all features normalized and the mean accuracy of 93.6% from logistic regression with all features normalized across 30 experiments for regression and classification, respectively. From all the experiments above, we observe that linear models perform better when doing both regression and classification



(a) logistic regression (test)



(b) logistic regression (train)

Fig. 13: Confusion Matrix (all Normalized Features)

works, to a small but statistically significant extent. Furthermore, the normalization does indeed improve both regression and classification works, however, the vector normalization harms the result of the neural network when doing regression. In addition, only using two chosen features leads to worse predictions and classification for both training and testing sets.

B. Limitation

The limitations of these models can be discussed separately for each of the three models. With regard to linear regression, the correlation coefficient between these data and the 'ring' (slightly greater than 0.5) indicates that there is no particularly strong linear relationship between them. The linear regression using only 'diameter' and 'shell weight' demonstrates a relationship between these two sets of data and 'ring' that appears to be logarithmic, as illustrated in the two dot plots. Further detailed processing of the data may be required. Additionally, further work is needed to identify the most relevant variables to improve its performance [12].

Concerning logistic regression, there is minimal scope for further improvement. The accuracy obtained is already very high. The same is true of LR, where the next step is to select the most appropriate features.

A significant limitation of neural networks is that the amount of data is insufficient to fully leverage their capabilities. In the course of our experiment, the sample size employed for training purposes was less than 2,500. The challenge of small data would necessitate a fundamentally different approach than that of big data. In the context of limited data, specifically when the sample size is below 5,000, the optimal depth and width for achieving maximum accuracy remain unclear [13]. To enhance the efficacy of the neural network, it is recommended to seek data with a greater quantity or to utilize more contemporary neural networks that are capable of adapting to limited data sets. Furthermore, in comparison to alternative models, neural networks are inherently time-consuming. The potential for optimization can be explored by testing additional optimizers, to enhance the efficiency of neural networks. For instance, Adam, Nesterov Accelerated Gradient (NAG), and AdaDelta are frequently employed to accelerate and stabilize the convergence [14]. In the end, for the choice of hyperparameter, in our experiment, we tested it artificially, which is pretty time-consuming, However, there is an algorithm in sklearn helps us to find the best hyperparameter more efficiently and accurate [15].

In terms of normalization, vector normalization scales the feature vector of each sample into a unit vector, which is more suitable for models with greater influence on the direction of feature vectors. In regression prediction tasks, if the actual size of the feature vector is important, using vector normalization may lose information and affect model performance, particularly for input vectors with extreme norms [16].

Assume we have two feature vectors, vector1 $= (x_1, x_2, x_3, \ldots, x_n)$, related to an age y, and vector2 $= (2x_1, 2x_2, 2x_3, \ldots, 2x_n)$, related to an age $2y$. After applying the normalization to both vectors, the results are:

$$\text{vector1}' = \text{vector2}' = \left(\frac{x_1}{\|\text{vector1}\|}, \frac{x_2}{\|\text{vector1}\|}, \ldots, \frac{x_n}{\|\text{vector1}\|} \right)$$

This shows that as vector normalization reduces the imbalance effect within feature vectors, it also reduces the effect of extreme norm samples on fitting results. Consequently, the information about significant differences is lost, which could exacerbate the Regression to the Mean phenomenon.

V. Conclusion

This paper examines the utility of linear regression, logistic regression, and neural networks for the prediction and classification of abalone age and also how to find the best hyperparameter for neural networks. it was determined that in the majority of instances, linear models demonstrate superior performance compared to neural networks on this data set. Nevertheless, we are currently employing the most conventional neural network and regression classification techniques. In future research, the use of additional novel neural networks may prove beneficial in addressing issues such as coping

strategies when the amount of data is limited. Concurrently, trying to find a better method to normalize the data may also improve the performance of models, as it still exists some limitations, only testing a few methods, when doing normalization. Moreover, the implementation of more contemporary regression and classification techniques, such as Support Vector Machines (SVM) (classification) [17], and pre-trained networks (new structures of neural networks to solve small data set problems) known as transfer learning, to initialize the neural network with the weights that have been trained in the relevant domain or domains and then to fine-tune the model with data that is pertinent to that domain [13], could potentially assist in resolving regression and classification challenges.

REFERENCES

[1] M. H. Haider, H. Zhang, S. Deivalaskhmi, G. Lakshmi Narayanan, and S.-B. Ko, "Is Neuromorphic Computing the Key to Power-Efficient Neural Networks: A Survey," in *Design and Applications of Emerging Computer Systems*, W. Liu, J. Han, and F. Lombardi, Eds., Springer Nature Switzerland, Cham, 2024, pp. 91–113. doi: 10.1007/978-3-031-42478-6_4.

[2] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. doi: https://doi.org/10.1016/0893-6080(89)90020-8.

[3] H. Osipyan, B. I. Edwards, and A. D. Cheok, "Deep Neural Network Applications," 1st ed., CRC Press, 2022. Available: https://doi.org/10.1201/9780429265686.

[4] G. I. Diaz, A. Fokoue-Nkoutche, G. Nannicini, and H. Samulowitz, "An effective algorithm for hyperparameter optimization of neural networks," *IBM Journal of Research and Development*, vol. 61, no. 4/5, pp. 9:1–9:11, 2017. doi: 10.1147/JRD.2017.2709578.

[5] Chengyuan Zhang, "Using Abalone's Physical Features to Predict its Age," *Highlights in Science, Engineering and Technology*, vol. 47, pp. 95–105, May 2023. doi: 10.54097/hset.v47i.8171.

[6] Seda Guney, Irem Kilinc, Alaa Ali Hameed, and Akhtar Jamil, "Abalone Age Prediction Using Machine Learning," in *Pattern Recognition and Artificial Intelligence*, C. Djeddi, I. Siddiqi, A. Jamil, A. A. Hameed, and I. Kucuk, Eds., Springer International Publishing, Cham, 2022, pp. 329–338. doi: 10.1007/978-3-031-04112-9_25.

[7] T. Ullmann, G. Heinze, L. Hafermann, C. Schilhart-Wallisch, D. Dunkler, and TG2 of the STRATOS initiative, "Evaluating variable selection methods for multivariable regression models: A simulation study protocol," *Plos One*, vol. 19, no. 8, p. e0308543, 2024. Public Library of Science, San Francisco, CA, USA.

[8] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986. Nature Publishing Group, UK, London.

[9] J. Fu, B. Wang, H. Zhang, Z. Zhang, W. Chen, and N. Zheng, "When and why momentum accelerates SGD: An empirical study," *arXiv preprint arXiv:2306.09000*, 2023.

[10] A. G. Barnett, J. C. van der Pols, and A. J. Dobson, "Regression to the mean: what it is and how to deal with it," *International Journal of Epidemiology*, vol. 34, no. 1, pp. 215–220, Aug. 2004. doi: 10.1093/ije/dyh299.

[11] D. Bashir, G. D. Montañez, S. Sehra, P. S. Segura, and J. Lauw, "An Information-Theoretic Perspective on Overfitting and Underfitting," in *AI 2020: Advances in Artificial Intelligence*, M. Gallagher, N. Moustafa, and E. Lakshika, Eds., Springer International Publishing, Cham, 2020, pp. 347–358. doi: 10.1007/978-3-030-64984-5_27.

[12] Yu Geng, Qin Li, Geng Yang, and Wan Qiu, "Linear Regression," in *Practical Machine Learning Illustrated with KNIME*, Springer Nature Singapore, 2024, pp. 45–97. doi: 10.1007/978-981-97-3954-7_3.

[13] Rhett N. D'souza, Po-Yao Huang, and Fang-Cheng Yeh, "Structural Analysis and Optimization of Convolutional Neural Networks with a Small Sample Size," *Scientific Reports*, vol. 10, no. 1, p. 834, 2020. doi: 10.1038/s41598-020-57866-2.

[14] W. Cheng, X. Yang, B. Wang, and W. Wang, "Unbiased quasi-hyperbolic Nesterov-gradient momentum-based optimizers for accelerating convergence," *World Wide Web*, vol. 26, no. 4, pp. 1323–1344, Jul. 2023. doi: 10.1007/s11280-022-01086-3.

[15] H. Mendoza, A. Klein, M. Feurer, J. T. Springenberg, and F. Hutter, "Towards Automatically-Tuned Neural Networks," in *Proceedings of the Workshop on Automatic Machine Learning*, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., vol. 64, Proceedings of Machine Learning Research, New York, New York, USA: PMLR, Jun. 2016, pp. 58–65. Available: https://proceedings.mlr.press/v64/mendoza_towards_2016.html.

[16] S. Yue, P. Li, and P. Hao, "SVM classification: Its contents and challenges," *Applied Mathematics-A Journal of Chinese Universities*, vol. 18, no. 3, pp. 332–342, Sep. 2003. doi: 10.1007/s11766-003-0059-5.

[17] M. M. Ahsan, M. A. P. Mahmud, P. K. Saha, K. D. Gupta, and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," *Technologies*, vol. 9, no. 3, p. 52, 2021. MDPI.