Z5427959

Zhao Ruilin

Abalone Dataset:

Predicting Ring age Using Linear Regression&Neural Networks

Introduction:

The Abalone Dataset is a valuable resource for understanding the factors that influence the age of abalones, a type of marine mollusk. In the field of marine biology, determining the age of abalones is traditionally accomplished through a labor-intensive process that involves cutting the shell, staining it, and then counting the number of rings under a microscope. This time-consuming method can be impractical, and there is a need to explore alternative, more accessible methods for age prediction.

Analysis to be done:

In this analysis, I will explore the Abalone Dataset and conduct linear regression to predict the age of abalones based on their physical characteristics. This predictive modeling approach offers a more efficient and non-invasive alternative to traditional age determination methods and can contribute to our understanding of these fascinating marine creatures.

Dataset:

The data set used here can be downloaded from https://archive.ics.uci.edu/ml/datasets/abalone This dataset offers a solution by providing a set of physical measurements of abalones that can be more easily obtained, such as length, diameter, height, and various weights, including whole weight, shucked weight (meat weight), viscera weight (gut weight), and shell weight. These measurements can potentially serve as predictors to estimate the age of abalones. The target variable for prediction is the number of rings, which, when modified by subtracting 1.5, represents the age in years.

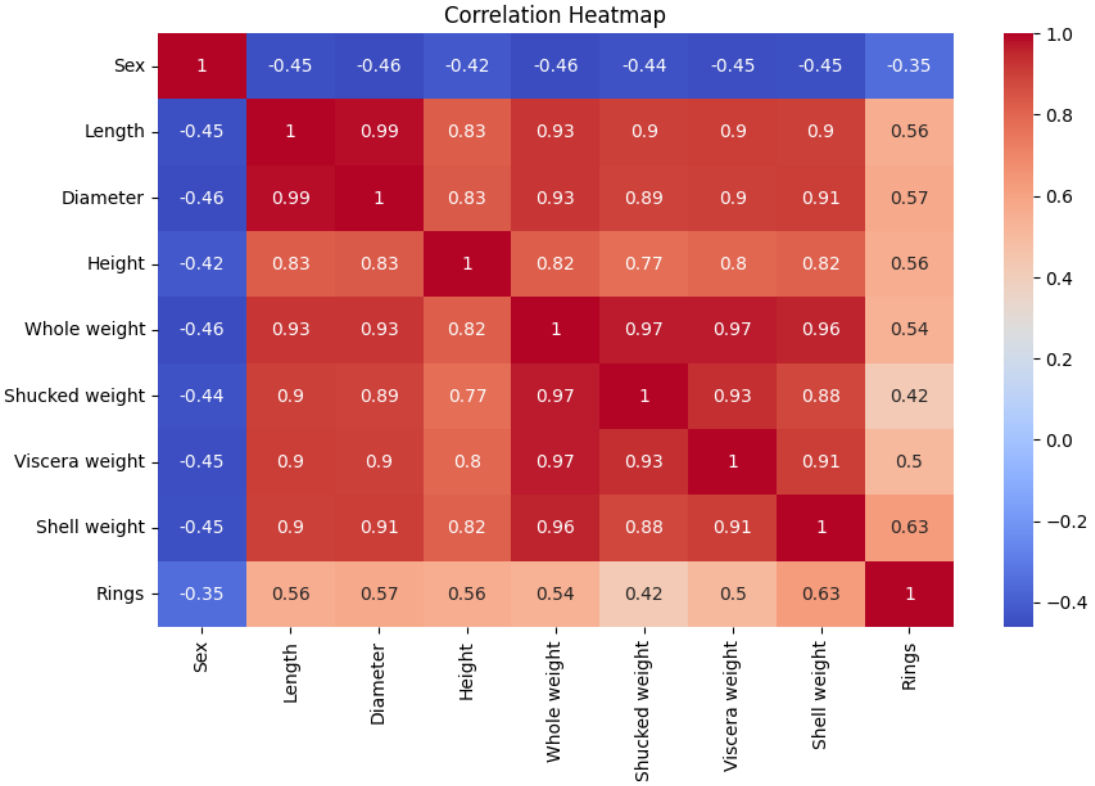
	0	1	2	3	4	5	6	7	8
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Data Processing

- Loading the data file, rename column names, convert classification values to numerical values, convert 'M' and 'F' to 0 and 1, and convert 'I' to 2.

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	0	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	0	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	1	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	0	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	2	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

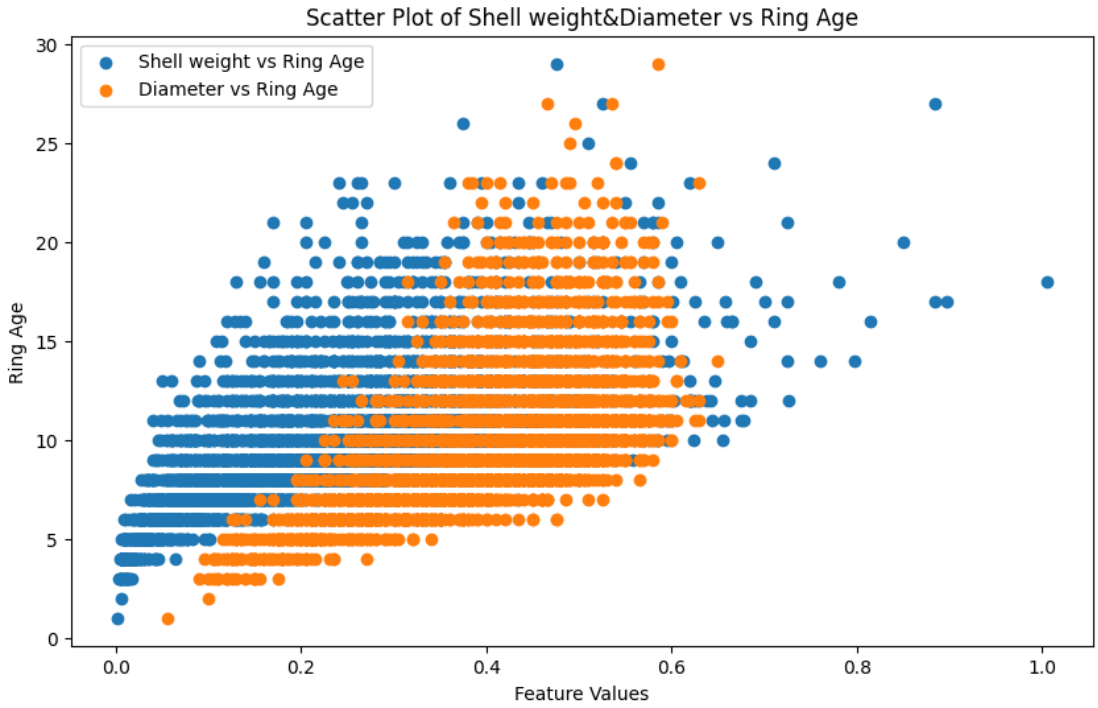
- Create a correlation heat map to understand the relationship between different features in the dataset. Determine which features are most relevant to the target variable 'ring age'.



Observations:

From the thermodynamic diagram, it can be seen that except for the feature sex, all other features exhibit a positive correlation with the target variable 'ring age'. Select the two most relevant features, 'Shell weight' and 'Diameter', for the following analysis.

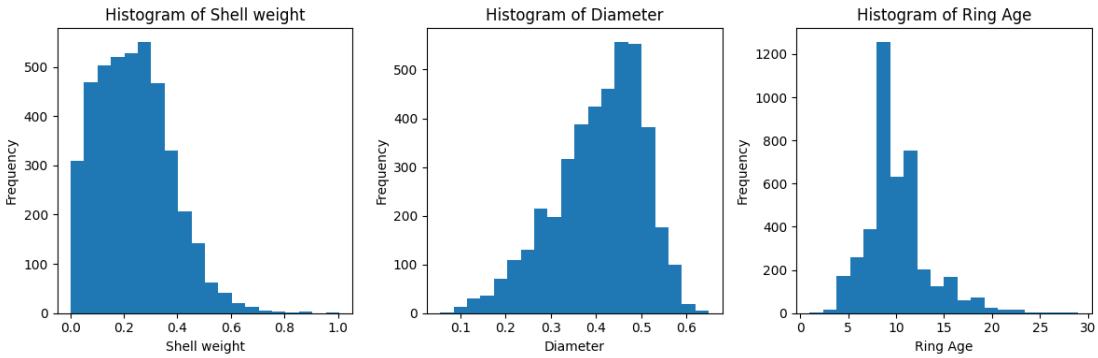
- Select the features' Shell weight 'and' Diameter 'to create scatter plots separately from the target variable' Rings'



Observations:

From the output results of the terminal, it can be seen that the features' Shell weight 'and' Diameter 'are positively correlated with the target variable' Rings' and exhibit a linear relationship.

- Select the features' Shell weight 'and' Diameter 'to create histograms separately from the target variable' Rings'



Observations:

From the figure, it can be seen that the feature Shell weight shows a left skewed distribution, while the feature 'Diameter' and the target variable 'Rings' exhibit a normal distribution.

- To prepare for modeling, a 60/40 training/testing split needs to be created. In addition, random seeds based on experimental numbers will be used to ensure repeatability.

Modelling: Linear regression model based on Sklearn

Step:

1) A function called LR has been defined, which takes the feature matrix X and the objective vector y as input parameters.

2) Internally, a linear regression model was initialized using LinearRegression().

3) Use a loop for multiple training and evaluation (30 times), each time using different random seeds to divide the dataset into training and testing sets.

4) In each loop, use train_test_split the dataset and fit the linear regression model with the training data.

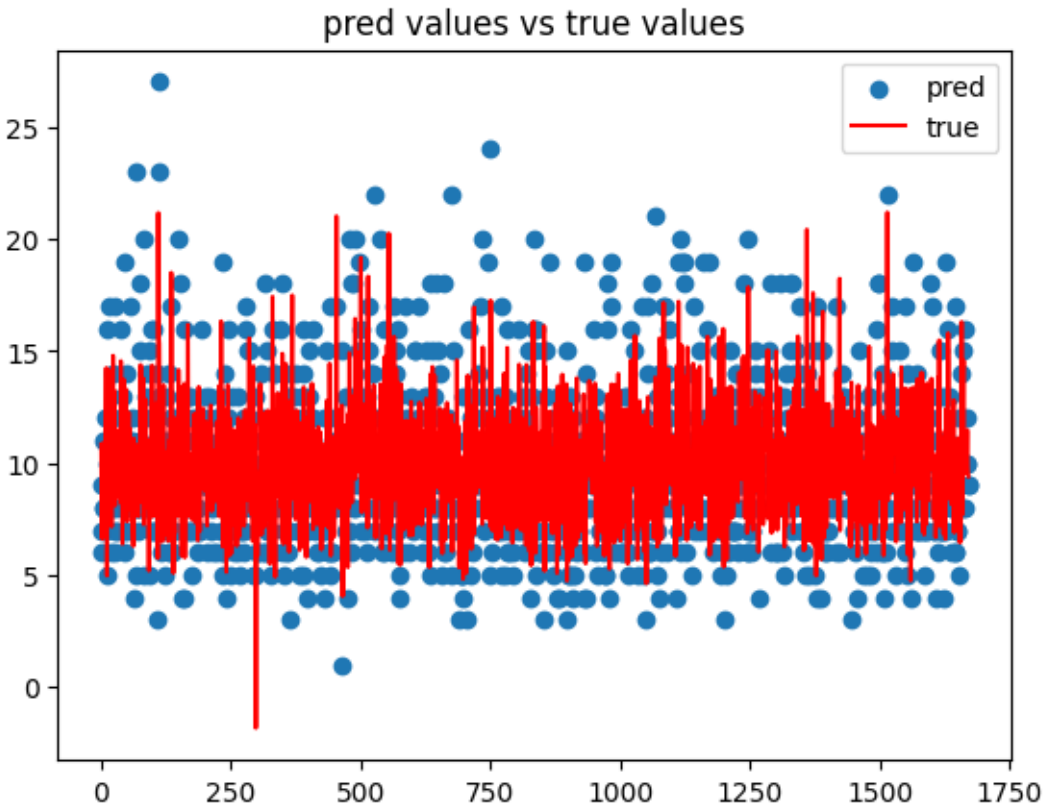
5) Calculate the root mean square error (RMSE) and determination coefficient (R) on the training and testing sets ²) To evaluate the performance of the model.

Evaluate the results of each cycle (RMSE and R ²) Add them to the corresponding list separately.

6) After the loop ended, RMSE and R were calculated on the training and testing sets ². Average value and standard deviation of, and print out the results. And visualize the regression results

Reporting Results

The performance evaluation indicators of the model are shown in the table below. From the results, it can be seen that the model performs well with a low standard deviation, indicating strong stability.



	score	Std
Average RMSE (Train)	2.18	0.03
Average RMSE (Test)	2.23	0.05
Average R-squared (Train)	0.54	0.01
Average R-squared (Test)	0.52	0.02

7) Normalize all features using the MinMaxScaler() normalization method, and compare the regression results with the untreated features mentioned above. From the comparison results, it can be seen that the normalized regression results are completely consistent with the untreated regression results, indicating that normalization does not affect the R-squared and RMSE of the model in linear regression

Without Normalization	score	Std	With Normalization	score	Std
Average RMSE (Train)	2.18	0.03	Average RMSE (Train)	2.18	0.03
Average RMSE (Test)	2.23	0.05	Average RMSE (Test)	2.23	0.05

Average R-squared (Train)	0.54	0.01	Average R-squared (Train)	0.54	0.01
Average R-squared (Test)	0.52	0.02	Average R-squared (Test)	0.52	0.02

8) Using the features' Shell weight 'and' Diameter 'to reestablish a linear model regression, the results are shown in the table below. From the table, it can be seen that when the features decrease, the values of the model evaluation indicators RMSE and R-squared decrease, and the standard deviation increases, indicating a decrease in the regression effect of the model. That is, when the features of the linear regression model decrease, the model performance will decrease

	score	Std
Average RMSE (Train)	2.50	0.04
Average RMSE (Test)	2.52	0.06
Average R-squared (Train)	0.40	0.01
Average R-squared (Test)	0.39	0.02

Modelling: Neural network model based on Tensorflow.keras

Step:

1. Defined a function named ‘create_model’, which is used to create a neural network model. The model is a sequential model consisting of two fully connected layers, with an optimizer of SGD and a loss function of mean square error.

2. A regression model was created using KerasRegression, with the construction function of create_ Model.

3. Defined a hyperparameter grid named ‘param_grid’, used to search for the optimal hyperparameter in grid search.

4. Four lists were initialized to store RMSE and R ^ 2 scores for training and testing.

5. A hyperparameter search was conducted using GridSearchCV, and the scoring criterion for the search was negative mean square error.

6. Divided the training set and test set, and then conducted a grid search on the training set.

7. The optimal hyperparameters were printed and the optimal model was obtained.

8. In a loop, the training and testing sets are re divided each time, and the model is trained on the optimal hyperparameters. The RMSE and R ^ 2 scores for the training and testing are calculated and stored.

9. The average and standard deviation of RMSE and R ^ 2 scores for training and testing were calculated and printed.

Reporting Results

linear regression	score	Std	Neural Network	score	Std
Average RMSE (Train)	2.18	0.03	Average RMSE (Train)	2.67	0.04
Average RMSE (Test)	2.23	0.05	Average RMSE (Test)	2.69	0.08
Average R-squared (Train)	0.54	0.01	Average R-squared (Train)	0.31	0.02
Average R-squared (Test)	0.52	0.02	Average R-squared (Test)	0.31	0.03

Further development

Comparing the linear regression model with the neural network model, it was found that the regression effect of the linear model was better than that of the neural network model

Okay, there are several possible reasons to speculate,

- The dataset has a small sample size, and neural networks may be prone to overfitting. Neural networks typically require more data to train to avoid overfitting. Linear models may be more stable on small datasets.
- Neural networks have many hyperparameters, such as number of layers, number of neurons, learning rate, etc. Setting these hyperparameters can be quite difficult. Linear models have relatively fewer hyperparameters, making them easier to adjust.
- Neural networks are not very sensitive to feature engineering and can learn to extract features from raw data. But linear models are more sensitive to the quality and selection of features. If feature engineering is not sufficient, the performance of linear models may be better.