Introduction

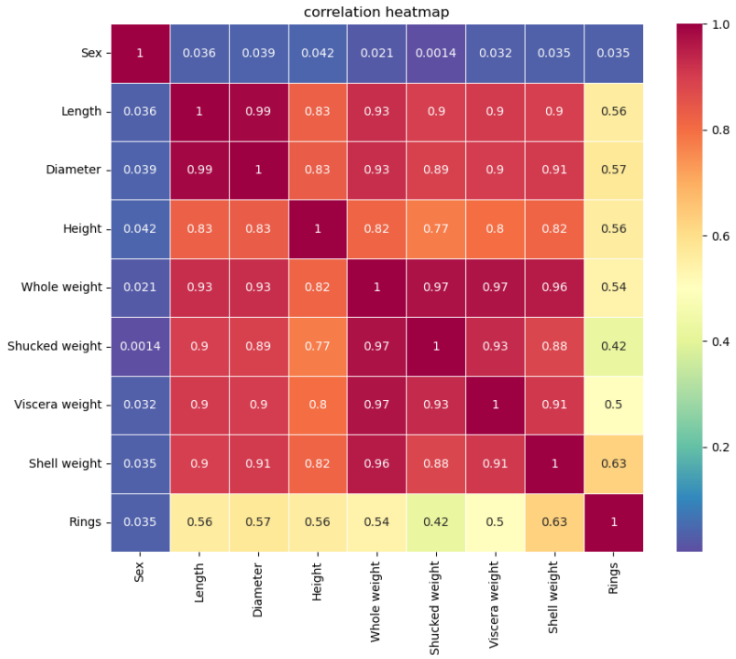
This report aims to discuss the observation of the data processing and the prediction of the ring age by the model.
In the training, I use random_state=88 + i, where i is equals to 0-2. By doing so, the losses are calculated from the same data, which make the result more fare. And the experiment time is 3.

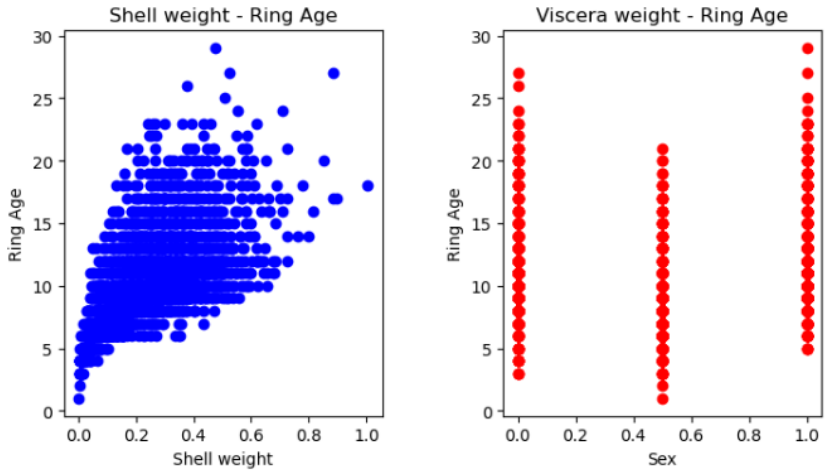
About the data

First of all, we check the dataset to confirm there are no null value in it and use the function map to mapping male into 0, female into 1 and infant into 2.

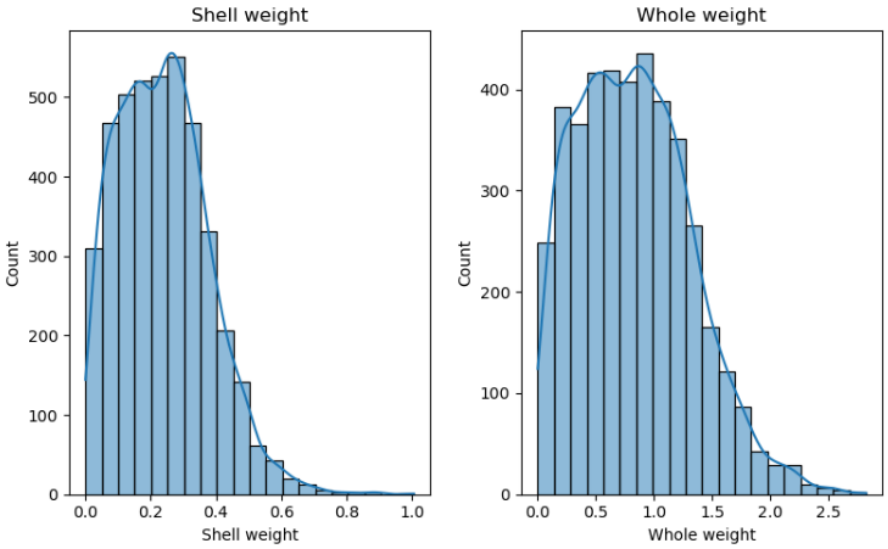
feature	type
Sex	Integer (0,1,2)
Length	float
Diameter	float
Height	float
Whole weight	float
Shucked weight	float
Shell	float
Rings(target)	Integer

After the mapping we can generate a heap map to identify the relationship between seven features. Obviously, the diagonal is all 1 because it means the relationship to itself. Moreover, we can know that shell weight and whole weight have most positive relationship with ring age while sex is the most irrelevant feature of all.





It is clear that the ring age tends to increase as the shell weight increase. However, whatever the sex is, the ring range chaoticly.

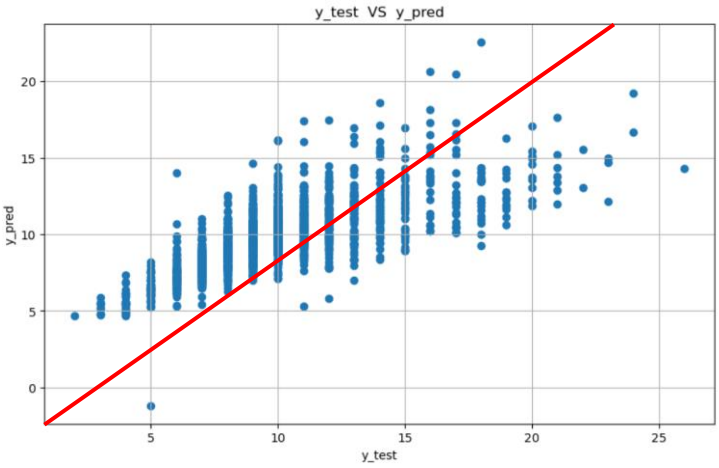


We create two histograms to show the most correlated feature shell weight and whole weight.

Finally, we split the training data and testing data by 60:40 in the model.

Modelling

Firstly, we use linear regression model to fit to the training set to predict the ring from all features (except “rings” which is the label).



We can see the although the prediction points are scattered, but it can almost fit in y=x, which means the prediction is fine.

Normalization:

```
Normalization:

RMSE: 2.229498090998581     vs     RMSE-normalization: 2.2294980909985807
R_squared_score: 0.5452831605614016     vs     R_squared_score-normalization: 0.5452831605614018

RMSE-normalization std: 0.03631080800488731
R_squared_score-normalization std: 0.004342783557588002
```

It is clear that the prediction is not goning any better after normalization. The reason might be the distribution of the data (features) are not normal distribution. Moreover, when I first try to normalize the data, I use standardscalar() but it failled, because there would have some null values in the data. But it is fine when I change the scalar into MinMaxScaler().

Selected shell weight and diameter:

```
Selected Shell weight and Diameter:

RMSE of Shell weight and Diameter: 2.5604996003386993, std: 0.03832495132597693
R_squared_score of Shell weight and Diameter: 0.40023524491673856
, std: 0.003834575108656393

Selected Shell weight and Whole weight:

RMSE of Shell weight and Whole weight: 2.4724393083480845
std: 0.04296142795892293
R_squared_score of Shell weight and Whole weight: 0.4407936799169552

std: 0.006226651912223755
```

As we select:

1. shell weight and diameter
2. shell weight and whole weight as our features.

The result seens not going better. Because the value of RSME raise up and the R square score become further to 1.

Neural network:

```
neural network:

Hidden Neurons: 20
Learning Rate: 0.01
RMSE: 3.266994893984465
R-squared: -0.0036360472432119817
```

We simply create a neural network to predict the rings. We set three number of hidden neuron (10,20,30) and three learning rate value (0.001, 0.01, 0.1) and put them into MLPRegressor() and corresponding library (scikit-learn) to choose the best parameter in this neural network. We finally get the best number of hidden neuron

10 and the value of learning rate is 0.01.
However, the value of RMSE and R square score do not improve.

Discussion

We build linear model and neural network on the data (with and without normalization), to compare the loss (RMSE and R2). By observing the loss, we can know which model and the method of data processing is suitable for this dataset.
In linear model, the value is acceptable, I first think the neural network would have a better result. However, is comes not that good. The reason that I think is that the dataset is still not complicated as it just has seven features (while the feature “sex” is not really correlate to the label) and thousands of data. So that we can simply use linear method to make our prediction. Moreover, there are some default implementation of Scikit-learn, which constraint the model.

Here are some observations on how to potentially improve the models:

1. Linear Regression Improvement: Linear regression uses a simple forward pass, and the weights and biases are not updated during training. To improve the model, we could consider using an iterative optimization algorithm, such as gradient descent, to update the model parameters. This way, the model can learn from the data and potentially has better results.

2. The split of the data: in this case, we use 60% of data for training and 40% of data for testing. We can try to have a validation.

3. Hyperparameter Tuning: For the neural network, you can try with different hyperparameters which can significantly impact the performance of the neural network, such as
a. number of layers
b. number of neurons
c. learning rate.

4. Dropout: Dropout is a technique used in neural networks to prevent overfitting. By dropping out a portion of neurons randomly during training, which can improve the model's generalization performance.

5. Consider Using Machine Learning tools: To build and train neural networks effectively, we can use machine learning frameworks like PyTorch(record gradient by default).

6. Runtimes: Since my computer is not good enough, so that I can just only try few numbers of test, normally, the loss can be lower if there are some more experiment times to be run.

In conclusion, the choice between linear regression and a neural network depends on the specific characteristics of the dataset and the problem we are trying to solve. It's essential to experiment with different approaches, preprocess the data appropriately, and fine-tune model parameters to achieve the best possible results.