CHAPTER 4
ANALYSIS AND DESIGN

4.1 Analysis

The data in cryptocurrency is fluctuative, dynamic, and does not have any trend pattern. Figure 1.1 and Figure 1.2 are the charts of BTC/USDT in 4 hours timeframe and ETH/USDT in 4 hours timeframe.

From the Figure 1.1 and Figure 1.2 outcome of the price is unpredictable and does not have any pattern, by this we can use the data as the sample to predict the situation of the product by date and sample for training.

Table 4.1: Data Table BTC/USDT

Date	Open	High	Low	Close	Volume
2017-8-17 15:0:0	4349.99	4485.39	4333.32	4427.3	62.67
2017-8-17 19:0:0	4427.3	4485.39	4333.42	4352.34	173.32
.....	.....	.....	.....	.....	.....
2018-11-9 3:0:0	6495.72	6515	6492.04	6495.38	432.74

The chart Figure 1.1 and Figure 1.2 was generated using the data table 4.1 but only using the values in close column. From the data above RSI value can be generated as an indicator which the value ranges from 0 to 100, by using the formula in Figure 4.1.

$$RSI = 100 - \frac{100}{1 + RS}$$

Figure 4.1: RSI Formula

The data will turn into a table consisting the columns of date and RSI this project will use the closing price for RSI formula shown in table 4.2.

Table 4.2: Data Table RSI

Date	RSI
2018-11-8 7:0:0	45.668956
2018-11-8 11:0:0	52.702906
2018-11-8 15:0:0	50.045027
2018-11-8 19:0:0	53.560731
2018-11-8 23:0:0	41.161616
.....	.....

In order to maintain the weights value in artificial neural network, data normalization is needed to process the data. The normalization will divide the RSI into three parts which are low, mid, high. Where the value of low RSI is below 30, for the mid between 30 and 70 and above 70 for high RSI, for more details look for the table 4.3.

Table 4.3: Data Table RSI details

RSI range	RSI value	RSI matrix
RSI < 30	Low	[0 0 1]
RSI >= 30 and RSI < 70	Mid	[0 1 0]
RSI >=70	High	[1 0 0]

4.2 Desain

Figure 4.2 is the flowchart of getting the data from the API and the process to normalize the data so it can be used in artificial neural network.

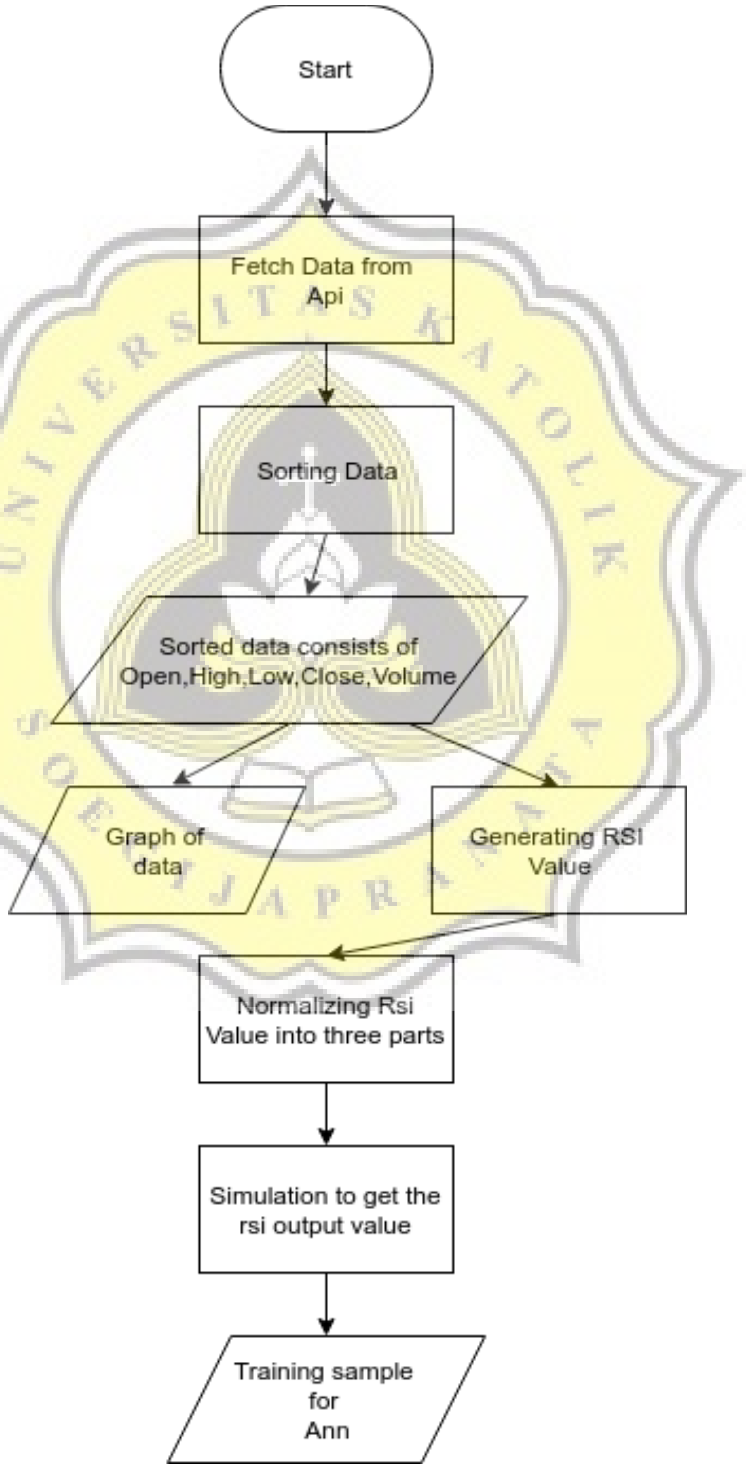

Figure 4.2: Data collecting and data normalizing flowchart

The program requests data from the cryptocompare API and the API returns the data, because in this project the programming language that used in collecting the data is nodejs and it is asynchronious the data returned will be returned in wrong order. In the second process we need to sort data in right timestamp order and then the graph of the data will show up to depic the fluctuativeness of the data.

The next step is to generate the RSI value from closing price. After the RSI value has been generated the data will still need to be normalized to be used for ANN. After normalized there will be simulation to get the output from the data which is whether 0 or 1.

Then the next stage is implementing artificial neural network and use the training sample that we get from the data normalization. Figure 4.3 is the flowchart of price predicting using artificial neural network algorithm.

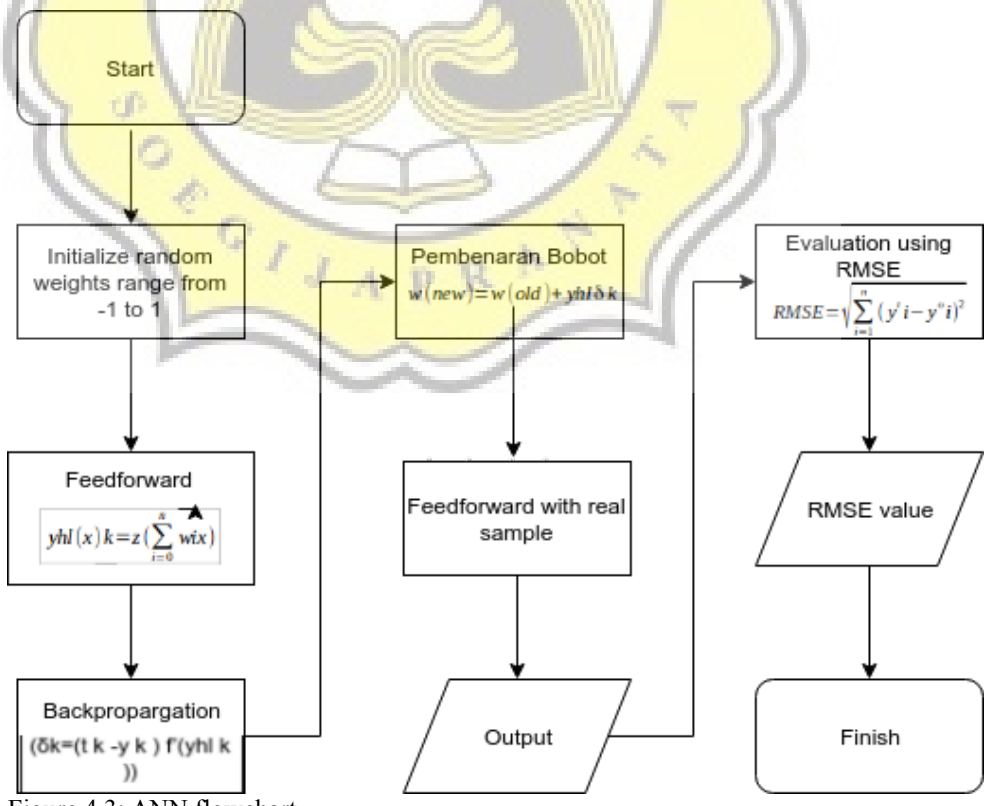

Figure 4.3: ANN flowchart

From the flowchart in Figure 4.3 the first step to do is to initialize the random weights which range from -1 to 1. The input sample which marked as x will be used for feedforwarding through the input layer to hidden layer with the formula in the flowchart above, then the value from output layer will show up.

```
[[ 0.42698174 -0.28678491  0.30628888  0.05361611]
 [ 0.2005165   0.63709148  0.59778815  0.00855487]
 [-0.47485116  0.5010164   0.04880367  0.77566845]]
```

Figure 4.4: Example of randomed weights matrix in python

The next stage is to learn to predict the output value by using backpropargation as the learning algorithm and then the correction of the weights by counting the delta and add the values to the old weights. The learning will takes time depends on the number of itterations or EPOCH.

After the learning stage, time to predict the outcome with the real data through feedforwarding based on the state of the RSI. Then the next step to evaluate the accuracy of the AI by using RMSE to evaluate all of the prediction output based on the real output.

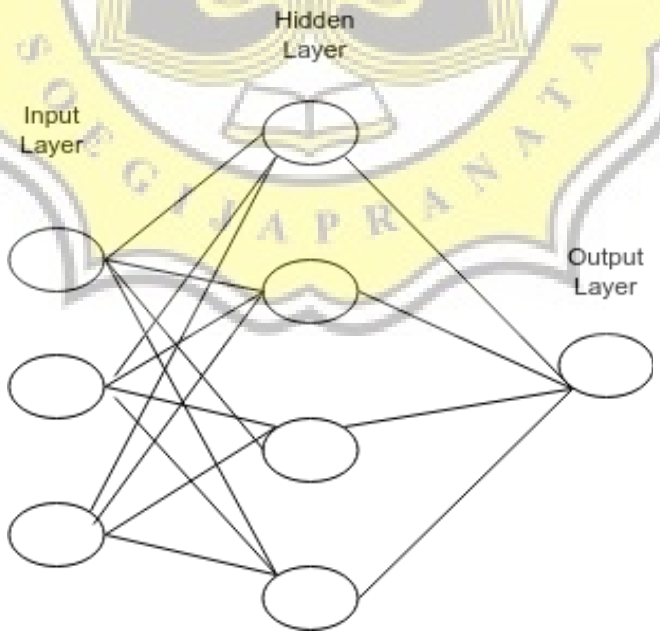

Figure 4.5: Neural Network Simple design

The Figure 2 shows the design of the neural network with 1 hidden layer which consists of 3x4 first weights and 1x4 for the second weights.