

CREDIT RISK PREDICTION USING NEURAL NETWORK BACKPROPAGATION ALGORITHM

Maradu Sihombing¹, Erwin D. Sitanggang², Maranata Pasaribu³, Misdem Sembiring⁴

^{1,2,3}Teknik Informatika, AMIK Medan Business Polytechnic, Medan, Indonesia

maradus44@gmail.com

Abstract

Article Info

Received : 21 August 2021

Revised : 10 October 2021

Accepted 04 November 2021

Credit is a business activity that contains high risk and greatly affects the health and survival of banking businesses and financing institutions. Predicting credit risk is very beneficial for banks or financing institutions in taking decisions to establish credit. Decision makers of banks and financing institutions must have the precautionary principle to minimize credit risk when credit will be provided. The study designed credit risk prediction software with artificial neural network *methods backpropagation* algorithms. Artificial neural network *backpropagation* with 1 *hidden layer* and the amount of data for training and testing as many as 20 pieces consisting of 5 models and using the logsig activation function is able to predict credit risk with a truth percentage of 70%-80%. Training and testing is used using matlab 6.1 *software*. Based on these results, the study recommends the development of artificial neural network algorithms as an effective method on credit risk prediction systems.

Keywords: Predictions, credit risk, Neural Networks, Backpropagation Algorithms

1. Introduction

Every company in the banking sector (Bank, BPR, CU, etc.) in providing credit / loans to customers trying to be selective as possible to approve the application for credit / loan so that the risk of congestion for repayment installments is estimated to be very small. Prospective customers must fill out a credit application form / loan with many rules / conditions, sometimes the impression makes it difficult for customers to get the loan. This is of course done by the company with the aim that in the repayment of the loan there is no problem until the expiration of the contract.

However, to ensure that the loan / loan is not bad will be very difficult to do, so it may take science and technology to be able to analyze and decide whether the loan application form / credit submitted is approved or not. To be able to find out the application for credit / loan is bad or can not be done scientifically by using artificial intelligence *Neural Networks (Neural Networks)* by knowing the patterns of prospective customer data. To recognize these patterns, backpropagation algorithms are used by training and testing data using a model of neural network architecture with multiple layers (*multilayer net*) consisting of input layers with 9 nodes, *hidden layers* with 3, 4, 5, 6 and 7 nodes and output layers (*output*) with 1 node.

The goal resulted from this study by using a neural network built using a reverse propagation algorithm (*Backpropagation*) with a *sigmoid* activation function. The activation function in neural networks is used to calculate the actual value of the output at the *hidden layer* and calculate the actual value of the output at the output *layer*. The desired results by conducting training and testing will be detected a value to determine whether the applicant's credit / loan is at risk or not. So that the decision is helped to decide whether to approve the credit / loan or not.

2. Method

3. 2.1 Introduction

Neural Networks can solve complex problems that are difficult or even impossible if solved using conventional computing. Currently Neural Networks are developing and have been pursued for various forms of research, among others.

1. Dwi Marisa Efendi, S.Kom., M.TI (2017), Application of Neural Network algorithm for creditworthiness prediction in Bina Bersama Lampung Utara cooperative, in this study was analyzed by the author using 10 patterns of determining creditworthiness namely salary, employment, family card, ktp, guarantee, UN, SIUP / SITU, loan length and installment length and average value. The analysis was conducted by using the Backpropagation algorithm with 9 input variables and X10 average values as output values with a pattern of 9-3-1 with three different learning rates of 0.1, 0.8 and 0.5 with MSE 0.000001, obtained the accuracy of achieving the desired target of about 88% with an average test time of 0.4 seconds.
2. Novan Wijaya (2018), Neural Network Model for credit risk evaluation, is useful for evaluating the possibility of credit proposed at risk of low-value or high-value bad luck judging from the output value (y). After several learning and credit risk evaluation tests, the best prediction results were 88%, with a hidden layer of 9, a learning speed value of 0.00935, a momentum value of 0.67 and a minimum error value of 0.0001.

2.2 Prediction

In general, predictions actually predict something that will happen in the future based on previous data. The result of the prediction is not 100% correct but close. Therefore, it is usually used to minimize a risk arising from a problem that occurs, not to avoid risk.

2.3 Credit risk

Credit risk that occurs in many financial institutions is the risk due to the inability of debtors to repay loans and pay interest, as well as a decrease in credit performance. The inability of the debtor occurs due to goodwill, disaster (force majeure) or due to the misunderstanding of credit officers in predicting repayment capacity, resulting in excess credit limit.

2.4 Neural Network

Artificial neural networks are one of the information processing systems designed by mimicking the workings of the human brain by doing the learning process through changes in the weight of its synapses. There are several applications of neural networks that are used for pattern recognition (e.g. letters, numbers, sounds or signatures) that have changed slightly. Neural networks can be used for forecasting by recognizing activities based on past data. The past data will be studied by neural networks so that they have the ability to make decisions about data that has never been studied (Anggriyani, 2015). Some network architectures that are often used in neural networks are single-screen networks, dual-screen and recurrent networks.

2.5 Artificial Neural Network Architecture

In JST, neurons are collected in layers (*layer*) called neuron layer (*neuron layers*). Neurons in one layer will be connected to the layers before and after. The information provided to the neural network will be propagated layer to layer, ranging from the input layer to the output layer through the hidden layer (*hidden layer*). There are 3 types of JST structure, namely: (Maharani Dessy Wulandari, 2012)

1. Single Layer network

Networks with a single layer have only one layer with a weight of connected weights. This network only receives inputs then will directly process them into output without having to go through a hidden layer.

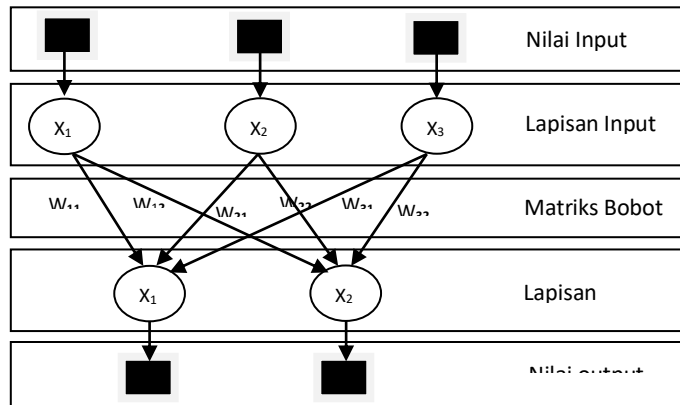


Figure 1. The input layer has 3 *neurons*

In Figure 2.3, the input layer has 3 *neurons*, namely X_1 , X_2 and X_3 . While in the output layer has 2 *neurons* namely Y_1 and Y_2 . *Neuronneuron*

The two layers are interconnected. How much the relationship between the two *neurons* is determined by the corresponding weight. All input units will be connected to each unit of output.

2. Plural Screen Network(*multi layer network*)

Networks with multiple layers have 1 or more layers located between the input layer and the output layer (having 1 or more hidden layers). Generally, there is a layer of weights located between 2 adjacent layers. Each inputted value is multiplied by the weight connected to each *neuron* in the hidden layer, then summed. The sum result is inputted on the activation function that occurs in the *neurons* of the layer

Hidden to get the results. Then, the result value of each hidden layer *neuron* is multiplied by the weight connected to each *neuron* on the output side. The sum result is entered in the activation function that applies to get the output value. Networks with multiple layers can solve more difficult problems than single-layer layers, of course with more complicated learning.

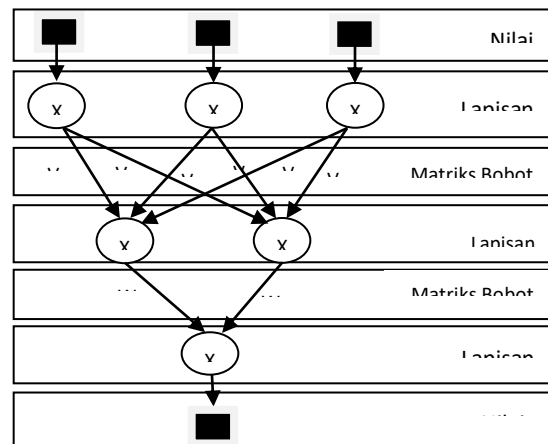


Figure 2. Plural Layer Architecture Images

3. Network With Competitive Layer (*Competitive Layer Network*)

It is a type of neural network that has a predetermined weight and has no training process. It is used to determine the winning *neurons* of a number of existing *neurons*. The weight value for each *neuron* is 1, while for other *neurons* it is a negative random weight.

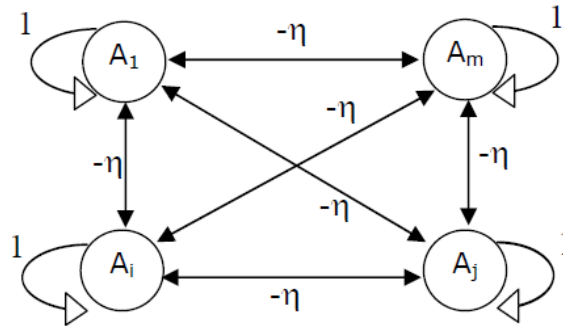


Figure 3. Competitive Layer Network

4. Backpropagation

According to Febrianto and Mustafidah (2013), backpropagation is an inherited learning algorithm and is usually used by perceptrons with multiple layers to change the weights connected to neurons in their hidden layers. Sangadji explained that the Backpropagation algorithm includes supervised training methods and is designed for operations on multi-layer feed forward networks. This algorithm is also used in settings applications because the training process is based on simple relationships. If the output gives the wrong result, then the weighing weight (W) is corrected so that the error can be reduced and the network response is expected to approach the target result and improve the weighing weight of the balancer on the hidden layer. Backpropagation Neural Network learning algorithm formulated by Puspasari and Sucipta in the journal Yustriana Handika Siregar (2017) is briefly: 1. Initialization of weights that can be done randomly. 2. Each neuron calculates the activation value of the input it receives. At the input layer, the activation value is the identity function. In hidden layers and outputs, activation values are calculated through the activation function. 3. Weight adjustment is affected by the magnitude of the error value between the output target and the current network output value. 4. Iterations will continue until certain minimum error criteria are met.

3. Results and Discussion

3.1 Variable Analysis of Borrower Determination

The eligibility assessment variable of prospective debtors is a criterion that becomes a reference in decision making on assessment using neural networks. The analysis will be grouped into 9 criteria, namely:

Table 1. List of Criteria in Determining eligibility of prospective borrowers

No.	Criterion	Information	Weight
1	X1: Marital Status	Marry	1
		Unmarried	2
		Widow/Widower	3
2	X2: Gender	Man	1
		Woman	2
3	X3: Guarantee	House/Land	1
		4-Wheeled Vehicle	2

		2-Wheeled Vehicles	3
		Savings/Deposits in BPR-DP	4
		Other	5
4	X4: Home Ownership	Own	1
		Family Property	2
		Rent	3
		Agency/Service	4
		MORTGAGE	5
5	X5: Work	Civil Servants	1
		Professional	2
		Self employed	3
		Pensioner	4
		Private Employees	5
		BUMN employees	6
		Housewife	7
		Other	8
6	X6: Salary	Divide the data by the maximum value of sample salary data	
7	X7: Couple's Salary	Divide the data by the maximum value of sample salary data	
8	X8: Age	Divide the data by the maximum value of sample salary data	
9	X9: Loan Tempo	6 months	1
		12 months	2
		18 months	3
		24 months	4
		>24 months	5

3.2 Input and Target Data

The process of determining inputs (inputs) and targets or outputs is done first for the data processing process, so it will make it easier to share data and data processing processes to get results in accordance with what was previously expected.

1) Input Data

Input data is obtained from the credit application application form and files completed by the applicant at PT Bank Perkreditan Rakyat Duta Paramarta medan.. Data from the credit application application form filled out by the applicant determined input variables that affect credit risk (in this case the authors refer to the journal Credit Risk Prediction with a backpropagation artificial neural network, SNATI 2007). In the case of determining input variables the author compares the journal with the credit application application application form on PT. People's Credit Bank Ambassador Paramarta. In conclusion, the input variable obtained as many as 9 pieces as in table and 1 variable target / output (1 for risky credit and 0 for non-risk credit).

2) Target /Output Data

The target/output data is 1 showing the credit, while 0 indicates the credit is not at risk. The target/output data can be seen in table 3

Table 2. Data Target

No.	Information	Weight
-----	-------------	--------

1	Risky credit	1
2	Credit is not risky	0

3.2.1 Data Processing

The data processing that will be done is to change the credit application data on PT. Bank Credit Rakyat Duta Paramarta Medan based on the value of representation in table 1, From the results of data processing carried out, the credit application data on PT. The People's Credit Bank Duta Paramarta Medan is processed as contained in Table 3 below:

Table 3. Results of Credit Application Data Representation PT. People's Credit Bank Duta Paramarta Medan

No	X1	X2	X3	X4	X5	X6	X7	X8	X9	Target
1	1	1	1	1	5	0,22	0	0,3667	2	0
2	1	1	3	2	5	0,3	1	0,45	2	1
3	1	1	1	1	3	0,45	0	0,8333	3	0
4	1	1	3	1	8	0,15	0	0,6167	2	0
5	1	2	1	1	3	0,525	0	0,6167	4	0
6	1	1	1	1	5	1	0	0,4167	4	0
7	1	1	5	1	3	0,2	0	0,8667	2	0
8	1	1	1	1	3	0,4	0	0,8333	2	0
9	1	1	2	1	8	1	0,7	0,5	5	0
10	1	2	1	1	3	0,65	0	1	5	0

3.2.2 Data Transformation

In the *backpropagation* algorithm using the *sigmoid* function, the data displayed must be transformed first because the output range of the *sigmoid* activation function is (0 and 1). Data can be transformed into intervals (0 and 1), but it will be better at intervals (0.1, 0.9). This is because the *sigmoid* function is an *asymptotic* function whose value never reaches (0 or 1), using the formula (1).

Table 4. Results of Credit Application Data Transformation pt. People's Credit Bank Duta Paramarta Medan

No.	X1	X2	X3	X4	X5	X6	X7	X8	X9	Target
1	0,100	0,100	0,100	0,10	0,420	0,1659	0,10	0,1000	0,1000	0
2	0,100	0,100	0,500	0,50	0,420	0,2412	0,900	0,2052	0,1000	1
3	0,100	0,100	0,100	0,100	0,100	0,3824	0,100	0,6894	0,3667	0
4	0,100	0,100	0,500	0,100	0,900	0,1000	0,100	0,4158	0,1000	0
5	0,100	0,900	0,100	0,100	0,100	0,4529	0,100	0,4158	0,6333	0
6	0,100	0,100	0,100	0,100	0,420	0,9000	0,100	0,1632	0,6333	0
7	0,100	0,100	0,900	0,100	0,100	0,1471	0,100	0,7316	0,1000	0
8	0,100	0,100	0,100	0,100	0,100	0,3353	0,100	0,6894	0,1000	0
9	0,100	0,100	0,300	0,100	0,900	0,9000	0,660	0,2684	0,9000	0
10	0,100	0,900	0,100	0,100	0,100	0,5706	0,100	0,9000	0,9000	0

3.3 Data Training with Models 9-3-1

Data to be processed using *matlab* with architectural patterns 9-3-1, with training data as many as 10 pieces from table 3

Table 5. Results and Errors of Training Input Data with Models 9 – 3 – 1

No.	Training Input Data									Target	JST 9-3-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error
1	0,100	0,100	0,100	0,10	0,420	0,1659	0,100	0,1000	0,1000	0	0.0000	0
2	0,100	0,100	0,500	0,50	0,420	0,2412	0,900	0,2052	0,1000	1	0.9935	0,0065
3	0,100	0,100	0,100	0,100	0,100	0,3353	0,100	0,6894	0,1000	0	0.0000	0
4	0,100	0,100	0,300	0,100	0,900	0,9000	0,660	0,2684	0,9000	0	0.0000	0
5	0,100	0,900	0,100	0,100	0,100	0,5706	0,100	0,9000	0,9000	0	0.0000	0
6	0,100	0,100	0,300	0,100	0,100	0,4576	0,100	0,2684	0,1000	0	0.0000	0
7	0,100	0,100	0,500	0,900	0,900	0,2882	0,420	0,7737	0,1000	1	0.9939	0,0061
8	0,100	0,100	0,100	0,100	0,100	0,1471	0,900	0,6474	0,6333	0	0.0000	0
9	0,100	0,100	0,100	0,100	0,420	0,3118	0,100	0,3105	0,1000	0	0.0000	0
10	0,100	0,100	0,100	0,100	0,100	0,1471	0,100	0,6263	0,6333	0	0.0000	0

3.4 Data Testing with Matlab Model 9-3-1

After training with architectural patterns 9-3-1, it continued with testing with training testing data as much as 10 pieces of data from tabel 3.3. The data testing steps using matlab are set as follows:

1. Enter the test data with different variables with input variables at the time of training.
2. Enter the test target data with variables that are different from the target variables at the time of training.
3. See the Resulting Output

To see the results issued by the network can use the following commands:

```
>>[a,Pf,Af,e,Perf]=sim(net,q,[],[],u)
```

Here are the test results with 10 test data with a 9-3-1 testingpattern. The test results data can be seen in table 6 as follows:

Table 6. Results and Errors of Data Input Testing with Models 9 –3– 1

No.	Training Input Data									Target	JST 9-3-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error
1	0,500	0,100	0,500	0,500	0,420	0,4529	0,900	0,3105	0,6333	1	0.9737	0.0263
2	0,100	0,100	0,100	0,100	0,100	0,3824	0,100	0,6894	0,3667	0	0.0000	0.0000
3	0,100	0,100	0,500	0,100	0,900	0,1000	0,100	0,4158	0,1000	0	0.0307	- 0.0307
4	0,100	0,900	0,100	0,100	0,100	0,4529	0,100	0,4158	0,6333	0	0.0000	0.0000
5	0,100	0,100	0,100	0,100	0,420	0,9000	0,100	0,1632	0,6333	0	0.0000	0.0000
6	0,100	0,100	0,900	0,100	0,100	0,1471	0,100	0,7316	0,1000	0	0.0000	0.0000
7	0,100	0,100	0,100	0,100	0,100	0,9000	0,260	0,4790	0,3667	0	0.0000	0.0000
8	0,100	0,100	0,500	0,100	0,420	0,1471	0,340	0,4790	0,6333	0	0.0000	0.0000
9	0,100	0,900	0,100	0,100	0,100	0,3353	0,100	0,5210	0,6333	0	0.0000	0.0000
10	0,100	0,900	0,100	0,100	0,100	0,9000	0,100	0,8579	0,9000	0	0.0000	0.0000

Table 7. Data Training with Model 9-4-1

No.	Training Input Data									Target	JST 9-4-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error
1	0,50	0,10	0,50	0,50	0,42	0,4529	0,90	0,3105	0,6333	1	0.9669	0.0331
2	0,10	0,10	0,10	0,10	0,10	0,3824	0,10	0,6894	0,3667	0	0.0060	-0.0060
3	0,10	0,10	0,50	0,10	0,90	0,1000	0,10	0,4158	0,1000	0	0.0040	-0.0040
4	0,10	0,90	0,10	0,10	0,10	0,4529	0,10	0,4158	0,6333	0	0.0029	-0.0029
5	0,10	0,10	0,10	0,10	0,42	0,9000	0,10	0,1632	0,6333	0	0.0106	-0.0106
6	0,10	0,10	0,90	0,10	0,10	0,1471	0,10	0,7316	0,1000	0	0.0106	-0.0058
7	0,10	0,10	0,10	0,10	0,10	0,9000	0,26	0,4790	0,3667	0	0.0140	-0.0140
8	0,10	0,10	0,50	0,10	0,42	0,1471	0,34	0,4790	0,6333	0	0.0081	-0.0081
9	0,10	0,90	0,10	0,10	0,10	0,3353	0,10	0,5210	0,6333	0	0.0024	-0.0024
10	0,10	0,90	0,10	0,10	0,10	0,9000	0,10	0,8579	0,9000	0	0.0032	-0.0032

Table 8. Data Testing with Matlab Model 9-4-1

No.	Training Input Data									Target	JST 9-4-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error
1	0,10	0,10	0,10	0,10	0,42	0,1659	0,10	0,1000	0,1000	0	0.0055	-0.0055
2	0,10	0,10	0,50	0,50	0,42	0,2412	0,90	0,2052	0,1000	1	0.9295	0.0705
3	0,10	0,10	0,10	0,10	0,10	0,3353	0,10	0,6894	0,1000	0	0.0061	-0.0061
4	0,10	0,10	0,30	0,10	0,90	0,9000	0,66	0,2684	0,9000	0	0.6101	-0.6101
5	0,10	0,90	0,10	0,10	0,10	0,5706	0,10	0,9000	0,9000	0	0.0020	-0.0020
6	0,10	0,10	0,30	0,10	0,10	0,4576	0,10	0,2684	0,1000	0	0.0103	-0.0103
7	0,10	0,10	0,50	0,90	0,90	0,2882	0,42	0,7737	0,1000	1	0.0037	0.9963
8	0,10	0,10	0,10	0,10	0,10	0,1471	0,90	0,6474	0,6333	0	0.0295	-0.0055
9	0,10	0,10	0,10	0,10	0,42	0,3118	0,10	0,3105	0,1000	0	0.0058	-0.0058
10	0,10	0,10	0,10	0,10	0,10	0,1471	0,10	0,6263	0,6333	0	0.0056	-0.0056

Table 9. Data Training with Model 9-5-1

No.	Training Input Data									Target	JST 9-5-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error
1	0,50	0,10	0,50	0,50	0,42	0,4012	0,10	0,1632	0,1000	1	0.9999	0.0001
2	0,10	0,10	0,10	0,10	0,42	0,2176	0,10	0,3105	0,1000	0	0.0003	-0.0003
3	0,10	0,10	0,10	0,10	0,42	0,2882	0,10	0,3105	0,3667	0	0.0002	-0.0002
4	0,50	0,10	0,50	0,50	0,42	0,2882	0,10	0,3105	0,6333	0	0.0022	-0.0022
5	0,10	0,10	0,10	0,10	0,10	0,1941	0,10	0,5842	0,3667	1	0.9921	0.0079
6	0,10	0,10	0,50	0,10	0,42	0,3824	0,10	0,6894	0,6333	0	0.0013	-0.0013
7	0,10	0,90	0,10	0,10	0,10	0,3353	0,10	0,5210	0,6333	1	0.9739	0.0261
8	0,10	0,90	0,10	0,10	0,10	0,2412	0,10	0,7948	0,9000	0	0.0008	-0.0008
9	0,10	0,10	0,50	0,10	0,10	0,1659	0,10	0,3736	0,1000	0	0.0130	-0.0130
10	0,10	0,10	0,10	0,10	0,42	0,2412	0,26	0,5210	0,6333	0	0.0134	-0.0134

Table 10. Pengujian Data dengan Matlab Model 9-5-1

No	Data Input Pelatihan									Target	JST 9-5-1	
	X1	X2	X3	X4	X5	X6	X7	X8	X9		Output	Error

										et		
1	0,10	0,10	0,10	0,10	0,42	0,1471	0,58	0,7316	0,3667	0	0.0003	-0.0003
2	0,50	0,10	0,50	0,50	0,42	0,3118	0,10	0,1632	0,6333	1	0.9999	0.0001
3	0,10	0,10	0,10	0,10	0,10	0,1471	0,10	0,5631	0,3667	0	0.0002	-0.0002
4	0,10	0,10	0,50	0,10	0,42	0,4012	0,50	0,3736	0,6333	0	0.0022	-0.0022
5	0,10	0,90	0,10	0,10	0,10	0,2176	0,10	0,5210	0,6333	0	0.9921	0.0079
6	0,10	0,90	0,10	0,10	0,10	0,2882	0,10	0,8579	0,9000	0	0.0013	-0.0013
7	0,10	0,10	0,50	0,10	0,10	0,2882	0,10	0,6053	0,1000	0	0.9739	0.0261
8										0		-0.0008
	0,10	0,90	0,10	0,10	0,42	0,1941	0,50	0,5631	0,1000		0.0008	
9	0,10	0,10	0,10	0,10	0,10	0,3824	0,10	0,3736	0,3667	0	0.0130	-0.0130
10	0,10	0,10	0,10	0,10	0,42	0,3353	0,10	0,1632	0,3667	0	0.0134	-0.0134

Table 11. System Test Results

No.	Testing with Matlab Model 9-3-1			Testing with Matlab Model 9-4-1			Testing with Matlab Model 9-5-1		
	Target	Output	Error	Target	Output	Error	Target	Output	Error
1	1	0.9737	0.0263	0	0.0055	-0.0055	0	0.0003	-0.0003
2	0	0.0000	0.0000	1	0.9295	0.0705	1	0.9999	0.0001
3	0	0.0307	-0.0307	0	0.0061	-0.0061	0	0.0002	-0.0002
4	0	0.0000	0.0000	0	0.6101	-0.6101	0	0.0022	-0.0022
5	0	0.0000	0.0000	0	0.0020	-0.0020	0	0.9921	0.0079
6	0	0.0000	0.0000	0	0.0103	-0.0103	0	0.0013	-0.0013
7	0	0.0000	0.0000	1	0.0037	0.9963	0	0.9739	0.0261
8	0	0.0000	0.0000	0	0.0295	-0.0055	0	0.0008	-0.0008
9	0	0.0000	0.0000	0	0.0058	-0.0058	0	0.0130	-0.0130
10	0	0.0000	0.0000	0	0.0056	-0.0056	0	0.0134	-0.0134

Of the three models created in the process of testing the Neural Network system with the Backpropagation algorithm using matlab with a sample of data as much as 10 credit applicant data, it was concluded that the 9-3-1 model produced the best MSE.

4. Conclusions

Backpropagation Artificial Neural Network can be developed as an alternative method of developing an effective credit risk prediction system, The *backpropagation* algorithm cannot ascertain the length of iteration required and how much *epoch* must be passed to achieve the met *goal*. Therefore, it is necessary to do re-research to determine the parameters of the network are made so as to produce a relatively fewer number of iterations. Artificial Neural Network (JST) backpropagation with 1 hidden layer that has been trained with 20 pieces of data is able to predict credit risk with a truth percentage of 70%-80%. The more data will be trained will require many iterations carried out until it reaches the desired target, so that to speed up the iteration process, the learning rate is enlarged to have a value of 0.3. Of the 5 (five) JST models that have been done in accordance with existing data and the same amount of data (20 pieces of training and testing data) then the 9-3-1 model produces the best output and error.

Reference

- [1] Anggriyani, 2015. Klasifikasi Kanker Serviks Menggunakan Jaringan Syaraf Tiruan Backpropagation and Graphical User Interface (GUI), EISSN 265-2738.
- [2] Dwi Marisa Efendi, 2017. Penerapan Algoritma Jaringan Syaraf Tiruan untuk prediksi kelayakan kredit di koperasi Bina Bersama Lampung Utara, Prosiding ISSN 2598-0246, E-ISSN 2598-0238.
- [3] Febrianto dan Mustafidah 2013, Penerapan Jaringan Syaraf Tiruan dengan Metode Pembelajaran Backpropagation untuk mengetahui tingkat kualifikasi calon siswa pada sistem informasi penerimaan siswa baru di MAN 2, Banjarnegara. ISSN : 2086-9398.
- [4] Maharani Dessy Wulandari, Irawan Afrianto 2012, Perbandingan Metode Jaringan Syaraf Tiruan Learning Vector Quantization pada pengenalan wajah, edisi 1 Volume 1, 2012
- [5] Novan Wijaya, 2018. Model Jaringan Syaraf Tiruan untuk evaluasi resiko kredit, Journal of computer science and information systems, 2/1 (2018)
- [6] Yustria Handika Siregar, 2017. Prediksi perilaku pola jumlah mahasiswa menggunakan Jaringan Syaraf Tiruan dengan metode Backpropagation, Jurnal teknologi informasi vol 1 no 2 Desember 2017, e-ISSN 2615-2738
- [7] Yusuf Wira Nugraga 2017, Penerapan Jaringan Syaraf Tiruan Backpropagation pada Prediksi Penerimaan Kredit Nasabah, Vol 4 No. 1, ISSN 2615-5788
- [8] M. Rikzal Humam Al Kholili¹, Budi Darma Setiawan², Randy Cahya Wihandika³, Implementasi Metode Jaringan Saraf Tiruan Backpropagation Pada Prediksi Payload 4G di Telkomsel, Jember, Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X Vol. 3, No. 4, April 2019
- [9] Dwi Marisa Efendi, Penerapan Algoritma Jaringan Syaraf Tiruan Untuk Prediksi Kelayakan Kredit Di Koperasi Bina Bersama Lampung Utara, PROSIDING ISSN: 2598 – 0246 | E-ISSN: 2598-0238 SEMNAS IIB DARMAJAYA, 2017
- [10] Nur Hadian¹; Hafifah Bella Novitasari²; Ami Rahmawati, Klasifikasi Peminjaman Nasabah Bank Menggunakan Metode Neural Network, Jurnal PILAR Nusa Mandiri Vol. 15, No. 2 September 2019