FIT 5186 Assignment Paper

Potential Credit Card Default Clients Classification Using Neural Networks

Jing Chen (2819**)**
Shangyi Li (2819**)**

Submission date: 2017-05-28

Abstract

Default of credit card clients has long been an important issue in bank systems which may lead to huge loss. The purpose of this research is to predict the potential default status of clients efficiently and with high accuracy. In our experiment, we design and modify the architecture of the neural network to find an appropriate model to improve the performance of classify clients may default or not. Some features of clients, such as education level and gender are found having obvious influence on credit default. With our approach, the banks can establish a credit default detection mechanism, which can reduce risks and increase profits.

Key work: MFNN, 1-to-N encoding, SMOTE, f-measure

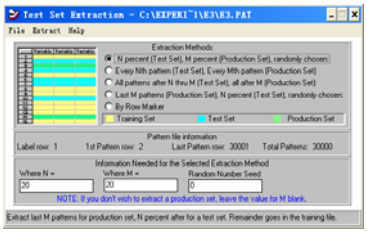
1. Introduction

Credit card issuers over issue cash and credit cards to unqualified applicants will lead to huge loss, because those unqualified cardholders may overuse their credit cards and accumulate heavy credit card debts (Yeh et al. 2009). It is important to find a credit assessment model to classify cardholders and judge they are default or not. How to use a client's past financial record and personal information design a model to judge the default is the main problem our group concerned. After research, we found that Baesens et al. (2003) and Yeh et al. (2009) used the neural network to do the classification experiments on eight real-life credit default data sets and got a good performance. Therefore, we want to find a suitable classification neural network model to classify clients could default or not by conducting a series of experiments.

We totally conduct 4 rounds contrast experiments. Experiment rounds 1 & 2 are utilized to choose the form of features and solve the imbalance dataset problem. Experiment rounds 3 & 4 are utilized to figure out the suitable neurons in hidden layer and the threshold of the model output. After four round experiments, we got an acceptable neuro network model to classify the default clients.

2. Data description and preprocessing

The data set of clients' information in our experiments is derived from UCI machine learning repository default of credit card client's data set. The data set has 30000 instances and 24 attributes. There is no value missing in the data set. To avoid overfitting, our group use 60% the data as the training data, 20% as the testing data and last 20% as the processing data. The data set divided result by neuroshell2 shows bellow:



This data set original total includes 23 attributes *X1-X23*, we only choose the *X1-X11* attributes as our experiments inputs because attributes *X12-X17* (*Amount of bill statement)* and *X18-X23* (*Amount of previous payment)* have already reflect in the attributes *X6-X11*. The detail of the descriptions of each original input attribute is given in Appendix Table 1.

The output of the network will be 0 (*Not default*) or 1 (*default*), which is the prediction of one's default probability. 1 means the person will default in the future, while 0 the opposite.

Because attributes *X3* (*education*) and *X4 (marriage)* have multi-classes, so we use *1-out-of-N* method, convert the single column categorical data into numerical data contains several columns with binary variables. In this paper, for example, we use a tool called Weka[1] to transform '1,0,0' to represent a client is married, '0,1,0' to represent a client is single and '0,0,1' to represent a client is divorced. In the same way, we use '1,0,0,0', '0,1,0,0', '0,0,1,0', '0,0,0,1' to represent the different education levels.

After we applied *1-to-N* method to attribute X3 and X4, the input of the MFNN model change from *11* to *16.*

After we use EXCEL make a simple descriptive statistical analysis, we know the training data set is imbalance, the output 0 (not default) instances is *3.25* times of the 1 (default) instances. We decide to use the Synthetic Minority Oversampling Technique (SMOTE) to balance the training dataset. Smote is a classical sampling algorithm based on oversampling techniques, oversampling on minority class (Chawla et al. 2002). We use the Weka tool to apply the SOMTE technique and make the training set has a relatively balance output class.

3. Training issues

3.1 The selection of architecture

The advanced system of Neuroshell2 offers 12 architectures implementing the backpropagation paradigm and 4 other architectures implementing other paradigms. According to the manual of neuroshell2, the data set with over 2000 patterns does not

[1] Weka is a tool that contains tools for data pre-processing, classification, and visualization, developed at the University of Waikato, New Zealand.

fit for the GRNN architecture. And after test the data under the beginner model, we found it will cost more than 30 mins to train the data. Therefore, we decide to experiment only on one Multilayer Feedforward Neural Networks (MFNN) model with different parameters changes. In all our experiments, we use *jump connections* backpropagation architecture in neuroshell2 with one hidden layer and 0.1 learning rate to train our model (*see Figure1*). We contrast the different neurons and threshold value in different experiments.

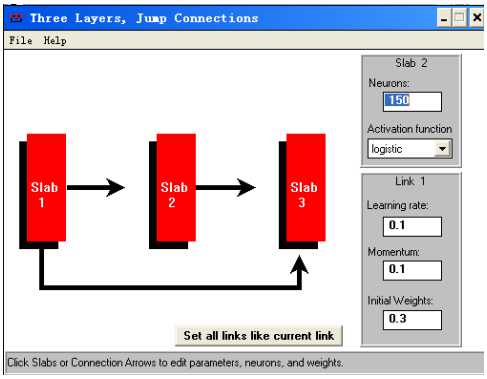


Figure 1 Jump connections architecture

3.2 The design of contrast experiment

First, in round 1 experiment, we choose the *jump connections* MFNN model with 150 neurons in hidden layer to contrast the influence of input features. We compare the experiment result of 11 original attributes inputs and the result of 16 inputs applied the 1-to-N encoding method which is introduced in *Section 2*.

Second, in round 2 experiment, we use the 11 features as inputs to prove the SMOTE method can improve the training result of MFNN model. If SMOTE method can improve the classification result, we can apply the method to following experiments.

Third, in round 3 experiment, according to the compare result of first step we choose the features with better result as the inputs of this round. We compare the different neurons on hidden layer to find a suitable training MFNN model under learning rate 0.1 and threshold 0.4. The basic hidden layer neuron nodes will be decided according to the formula below, based on this number every 50 neuron do an experiment.

Number of hidden neurons = 1/2 (Inputs + Outputs) + Sqrt (Number of Patterns)

At last, in round 4 experiment, we change the threshold value based on the result of round 3 experiment. We find the suitable threshold and hidden layer neurons of the MFNN model.

3.3 The selection of evaluation methods

Due to the severe imbalance of data, it makes more prediction classes biased toward majority class, that is class ‘0’ (not default). In this paper, we take measures to develop

a better model to make sure that the prediction result of the sample data, we use *f-score* which is the harmonic average calculated by the classification accuracy of class "1" and the recall rate, as a reference target of the improvement in prediction performance of the model.

We define the confusion matrix of our experiments as follows:

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**	
Actual result = 0 (not default)	True Negative (TN)	False Positive (FP)	
Actual result = 1 (default)	False Negative (FN)	True Positive (TP)	**Recall =TP / (TP+FN)**
accuracy = (TP+TN) / (TP+TN+FN+FP)		**Precision =TP / (TP+FP)**	

The definition of *f-score:*

f-score=(2*precision*recall) / (precision+recall)

f-score close to 1 indicates a near perfect prediction result. We write a *evaluate.py* python file to calculate the confusion matrix, accuracy and f-score, the code is included the submission file.

4. Result and analysis

We did four rounds experiments to find a suitable three layers jump connection MFNN model to classify the potential default clients. In our experiments, we decided to interrupt the training process when the minimum average error of the network has more than 100 epochs doesn't change. It took around 30-40 minutes to finish one model training experiment.

4.1 Experiment round 1

Experiment 1_1 with 11 features input, 1 output, and 150 hidden layer neurons;

Experiment 1_2 with 16 features input (apply 1-to-N method), 1 output, and 150 hidden layer neurons;

In both experiments the training datasets don't apply the SMOTE sampling method.

The confusion matrix of Experiment 1_1 is below, *f-score* = 0.473, *accuracy* = 0.82:

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4433	234
Actual result	848	485

= 1 (default)		

The confusion matrix of Experiment 1_2 is below, *f-score* = 0.503, *accuracy* = 0.825:

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4418	249
Actual result = 1 (default)	801	532

Analysis of round 1 experiment:

According to the f-score, we should apply the 1-to-N method to change the 11 features to 16 features as the input of the MFNN model in the Experiment round 3 & 4.

4.2 Experiment round 2

Experiment 2_1 with 11 features input, 1 output, and 150 hidden layer neurons, doesn't apply SMOTE method;

Experiment 2_2 with 11 features input, 1 output, and 150 hidden layer neurons, apply SMOTE method with 80% sampling rate;

The confusion matrix of Experiment 2_1 is below, *f-score* = 0.473, *accuracy* = 0.82:

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4433	234
Actual result = 1 (default)	848	485

The confusion matrix of Experiment 2_2 is below, *f-score* = 0.516, *accuracy* = 0.813:

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4278	389
Actual result = 1 (default)	733	600

Analysis of round 2 experiment:

According to f-score, the SMOTE method with 80% sampling rate can improve the model training result. So, apply the SMOTE to imbalance training dataset.

4.3 Experiment round 3

Experiment 3_1to Experiment 3_4 with the same threshold=0.4 the hidden layer neurons: 150, 200, 250,300. Calculate the f-score and accuracy, the result as below:

	Neuron =150	**Neuron =200**	**Neuron =250**	**Neuron =300**
F-score	0.503	0.512	0.526	0.486
Accuracy	0.825	0.825	0.816	0.822

Experiment 3_1 with 150 neurons confusion matrix

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4418	249
Actual result = 1 (default)	801	532

Experiment 3_2 with 200 neurons confusion matrix

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4398	269
Actual result = 1 (default)	782	551

Experiment 3_3 with 250 neurons confusion matrix

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4291	376
Actual result = 1 (default)	723	610

Experiment 3_4 with 300 neurons confusion matrix

Confusion matrix	**Predicted result = 0 (not default)**	**Predicted result = 1 (default)**
Actual result = 0 (not default)	4427	240
Actual result = 1 (default)	828	505

Analysis of round 3 experiment:

According to the highest f-score, we should choose the 250 neurons for hidden layer for further experiment in round 4 to find a suitable output threshold of the MFNN

model.

4.4 Experiment round 4

Based on round 3, use 250 neurons in hidden layer, experiment on different thresholds :0.2, 0.3,0.4,0.5,0.6. The result of apply the training model to processing sets as follow:

Experiment 4_1 with threshold=0.2 confusion matrix

Confusion matrix	Predicted result = 0 (not default)	Predicted result = 1 (default)
Actual result = 0 (not default)	3688	979
Actual result = 1 (default)	492	841

Experiment 4_2 with threshold= 0.3 confusion matrix

Confusion matrix	Predicted result = 0 (not default)	Predicted result = 1 (default)
Actual result = 0 (not default)	4098	569
Actual result = 1 (default)	621	712

Experiment 4_3 with threshold= 0.4 confusion matrix

Confusion matrix	Predicted result = 0 (not default)	Predicted result = 1 (default)
Actual result = 0 (not default)	4291	376
Actual result = 1 (default)	723	610

Experiment 4_4 with threshold= 0.5 confusion matrix

Confusion matrix	Predicted result = 0 (not default)	Predicted result = 1 (default)
Actual result = 0 (not default)	4387	280
Actual result = 1 (default)	779	554

Experiment 4_5 with threshold= 0.6confusion matrix

Confusion matrix	Predicted result = 0 (not default)	Predicted result = 1 (default)
Actual result = 0 (not default)	4482	185

Actual result = 1 (default)	858	475

	Threshold = 0.2	**Threshold = 0.3**	**Threshold = 0.4**	**Threshold = 0.5**	**Threshold = 0.6**
F-score	0.533	**0.545**	0.526	0.511	0.477
Accuracy	0.755	**0.80**	0.816	0.823	0.826

Analysis of round 4 experiment:

The jump connection MFNN model for this problem choose the threshold 0.3 can get the f-score is the highest. So, we can set the output greater than 0.3 to class '1'(default), and output smaller than 0.3 to class '0' (not default).

4.5 Analysis of whole experiments

After applying the SOMTE sampling method and the 1-to-N feature encoding method, the three-layer jump connection MFNN model, which has 250 neurons in hidden layer and output threshold equal to 0.3, return the best result with *f-score = 0.545* and *accuracy = 0.8* is a suitable model to classify the potential default clients.

Figure 2 is the relative contrition factors statistics information of the highest *f-score* return Experiment4_2, we can see the 16 inputs including personal information such as gender, age, and education level have strong relevance to the classification default result. The latest repayment status, the amount of the credit limitation and the education level have the relatively high contribution to influence the classification result.

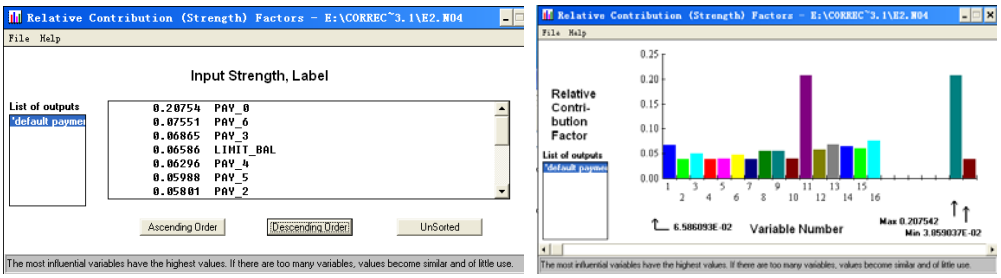


Figure 2 The relative contribution of features statistics of Experiment 4_2

5. Conclusion and limitations

Based on the above analysis, we find that in classify the clients may default credit card or not, combine features of clients, like education level, gender, age and credit limitation and repayment status, have obvious influence on credit default. According to our experiments, we can give out a feasible jump connection three layers MFNN model. The banks can reference our experiment result and establish a credit default detection mechanism, which can reduce the clients potential default risks.

The main limitation of our job is we don't consider the separate influence of each past financial record and personal information feature. Our experiments cannot give out the specific relevance of each single feature and the default result, in the future work can use the regression method to realize this purpose.

References

Baesens, B., Setiono, R., Mues, C., & Vanthienen, J. (2003). Using neural network rule extraction and decision tables for credit-risk evaluation. Management Science, 49(3), 312–329.

Chawla, N.V., Bowyer, K.W., Hall, L.O., & Kegelmeyer, W.P. (2002) SMOTE: Synthetic Minority Over-Sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.

Yeh, I. C., & Lien, C. H. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Systems with Applications, 36(2), 2473-2480.

Appendix

Table 1. Description of original attributes

X1	Amount of the credit limitation
X2	Gender of clients. Male = 1, while Female = 0.
X3	Education level of clients. Graduate school = 1, university=2, high school= 3 and others = 4.
X4	Marital status of clients. Married=1, single = 2, and others =3.
X5	Age of clients.
X6~X11	These six variables are the payment history from April to September 2005. For example, X6 is the repayment status in September. 2005, and X7 is the repayment status in August 2005. The measurement scale for the repayment status is: 0 = pay in time; 1 = payment delay for one month; 2 = payment delay for 2 months; …; 9 = payment delay for 9 months and above.