Analysis of Neural Network on Bank Marketing Data

Junming Zhang

College of Computer Science – ANU
108 Northbourne Avenue, Acton ACT 2601
u5722968@anu.edu.au

Abstract: This paper describes the implement of a typical neural network on a bank marketing data. We used both encrypted data and decrypted data to train the model, and comparisons were made from the results. We discussed about whether the decrypting technique improves the model. This paper also compared our method with another method that uses the same dataset. The comparisons were made and the results was discussed. By analyzing these comparisons, we made some conclusions and planned on the future work.

Keywords: neural network, classification, decrypting data, feature selection

1 Introduction

Neural network is a modern computer system to model the decision-making activities of human brain and nervous system. The two popular tasks for computer neural network are regression and classification.

As we all know the famous problem of back-propagation, which is to avoid encrypting the data that used by another user [1]. In this paper, we will focus on the input of a bank marketing dataset, trying to find out which information is redundant and what data needs to be encrypted. Here we built a standard neural network to solve the classification problem of predicting the clients' action based on the given data.

The raw data was called 'Bank Marketing' and was downloaded from the online Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets.html. The website currently maintains 436 data sets as a service to the machine learning community. This dataset holds 16 attributes of clients such as age, job, marital, education, etc. The dataset is typically made for classification problems, as the target value is a pair of binary values (yes / no).

The data contains many string type values, so in this paper we firstly replaced these value with numeric integers, and implement our neural network model to predict the outputs. Then using encoding techniques to decrypt the data in a reasonable way and let our model make the prediction based on the 'new 'decrypted data. We will see the comparison of accuracy from the results processing the data, and see whether the decryption actually improves the prediction.

2 Dataset Pre-processing

2.1 Choosing Dataset

In this paper, the dataset I chose is named 'Bank Marketing', and it's for solving a classification problem. It contains 17 attributes and 45211 instances which is runnable when implementing the neural network on my PC.

I chose this dataset for the following reasons:

1. Most importantly, I need to find a relevant paper to compare my result with. Since my goal is to improve the accuracy of prediction, so I need some paper that also uses this dataset while doing the predicting work as well.
2. There are a few abundant attributes that can be represented by one or several other attributes.
3. There are some attributes that need to be eliminated.

4. The data contains a lot of text values, which means it needs a lot of work to be done to encode the data, even in the early stage of preprocessing.
5. Marketing is one of the most popular topics nowadays, so if the encoding technique works well it could be very useful.

2.2 Massage the Data

There are in total 17 attributes in the dataset and most of them are not numeric values. The non-numeric values are shown below:

Inputs:

- **Job**: "admin.", "unknown", "unemployed", "management", "housemaid", "entrepreneur", "student", "blue-collar", "self-employed", "retired", "technician", "services".
- **Marital**: "married", "divorced", "single", "unknown".
- **Education**: "unknown", "secondary", "primary", "tertiary".
- **Default** (has credit in default?): "yes", "no".
- **Housing** (has housing loan?): "yes", "no", "unknown".
- **Loan** (has personal loan?): "yes", "no", "unknown".
- **Contact** (communication type): "unknown", "telephone", "cellular".
- **Month**: "jan", "feb", "mar", ..., "nov", "dec".
- **Poutcome** (outcome of the previous marketing campaign): "unknown","other","failure","success".

Output:

- Y (has the client subscribed a term deposit?): "yes", "no".

Here we simply assign the integer values to these string type data. (Note that we don't do much deeper analysis here in the first stage, because we need to see the whether there will be any difference between the encrypted data and decrypted data later. In other words, we are actually encrypting data by replace the raw data with numbers.)

Inputs:

Job	1	2	3	4	5	6	7	8	9	10	11	12
Marital	2	1	1	0								
Education	0	2	1	3								
Default/ Housing/ Loan	1	0	0									
Contact	0	1	2									
Month	1	2	3	4	5	6	7	8	9	10	11	12
Poutcome	0	0	-1	1								

Output:

Y	1	0

Figure 1 data-preprocessing

Reasons for assignments:

Jobs: Those are independent values that doesn't affect each other, so assigning them with different numbers from 1 to 12.

Marital: We would like to assign 2 to married people because it means at least two people are affecting each other. Here "1" was assigned to both single and divorced people because they are somehow very similar (both groups of people don't have any partners).

Education: Easy assignment based on the educated level. Sequentially, assign primary, secondary, tertiary with 1, 2 and 3. The number gradually increase when the education level is getting higher and higher.

Default/ Housing/ Loan: Binary inputs, so assign yes with 1 and assign no with 0. It is a standard and clear way of encoding binary inputs.

Contact: The two communication types are independent and don't affect each other, so just use different numeric numbers to represent them. Use 1 to represent telephone and use 2 to represent cellular.

Month: Simply use 1 to 12 to represent 12 months.

Poutcome: Here we assigned a negative number -1 to represent the penalty of failure, and use a positive 1 to represent the reward of success. 0 means some other situation that is neither success or failure, so we don't give any rewards or penalties based on that.

Y: Simple binary outputs. So again, assign 1 to yes and assign 0 to no.

Unknown values: We assign all of them with 0 as default value because the only number that can express no value is 0. 0 is more understandable than any positive or negative numbers.

3 Implementation of a neural network

3.1 Building a Neural Network

A two-layer neural network was implemented in this case. Pytorch was used to solve the classification problem. The detailed procedure is shown below:

Define the neural network:

- The neural network contains one input layer, one hidden layer and one output layer.
- Choose sigmoid function as our activation function.
- Since there are 16 attributes as inputs and 2 binary values as outputs, so we set 16 input neurons and 2 output neurons. Also, there will be 50 neurons in the hidden layer.

Loss Function: Here I used the Cross-Entropy loss to measure the performance of my classification model.

3.2 Evaluating the Neural Network

A confusion matrix is used for evaluating the performance of our classification model. In this case. We have a binary classifier that outputs yes/no. So, the confusion matrix method will be a relatively simple and clear method to use. It will list a 2×2 table, as shown below:

	Predicted: yes	Predicted: no
Actual: no	a	b
Actual: no	c	d

Figure 2 confusion matrix example

3.3 Decrypting the Data

Here I choose the method from the paper named *Decrypting Neural Network Data: A GIS Case Study. in Artificial Neural Nets and Genetic Algorithms* (*R.A. Bustos and T.D. Gedeon)*. To decrypt the data, we need to analyze the data in order to find out what data is meaningless or inappropriate. As I said before, the encoding method I used is simply replaced the string type values with numeric values, so it's actually causing the loss of information. However, now we need to modify some of the inappropriate value or even attributes based on our knowledge of bank marketing.

Enncoding decision:

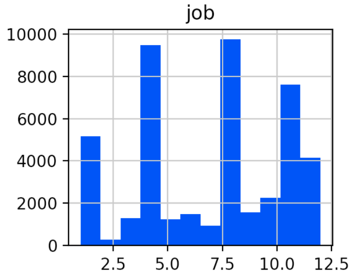

Figure 3 Job distribution

From the Figure above, we can tell that the values are not distributed mormally. There are 3 significant groups (above 6000), 2 medium groups, and others (lower or around 2000). So a single continuously input will not be appropriate. Here I used 6 inputs to distinguish between the popular types, medium types and rare ones. A table that decribes how it works is shown below:

	admin	unknown	unemployed	management	housemaid	entrepreneur	student	blue-collar	self-employed	retired	technician	services
H1				0.9								
H2								0.9				
H3											0.9	
M1	0.9											
M2												0.9
O		0.9	0.9		0.9	0.9	0.9		0.9	0.9		

Figure 4 Encoding table (All unmarked activations are 0.1)

4 Results and Discussion

4.1 Compare the encoded dataset with the original dataset

We are going to split the dataset contains 45211 instances into a training set with 40000 instances and a test set with 5211 instances. The 5211 instances for test data will be randomly chosen from the raw dataset. Here it is necessary to choose the test set randomly, because from my observation of the data, it does not distribute normally. For example, it's always 'unknown' as the value of the last attribute 'poutcome' for the first 24 thousand data. So, in order to train the data well using the unsatisfying distribution data, we must choose the training set and the test set randomly. After we split the data into training and test test set, we can start testing our model.

By running the model a few times, we found that when there were 20 hidden neurons, the model provided the best performance. Also, in order to balance the testing efficiency and testing quality, we choose 500 epochs. (When the epoch number is set to 600 or 400, the loss value increased.)

The table below shows the result after running both model.

METHOD	TIME COST	ACCURACY	LOSS
Original	39457.060ms	53.20481%	0.3311
Decrypted	41725.628ms	54.08672 %%	0.2966

Figure 5 The comparison between two results. (20 hidden neurons and 500 epochs are used.)

From the table, we can see that the time cost increased, which is understandable that we actually add 5 more attributes for training. The results are not satisfying as after the decrypting, it still only reached 54% accuracy. However, although the accuracy just improved a little, it doesn't mean the method is unnecessary. Here we have only encoded one of the attributes, and the accuracy improved almost 1%. Future analysis needs to be done in order to encode other attributes if necessary.

4.2 Compare with other works on the same data set

There is one paper that also uses the same dataset, which is called *A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems (*S. Moro, P. Cortez and P. Rita*)*. This binary classification task was modeled using a SVM algorithm that was fed with 26 attributes (after a feature selection step), using 2/3 randomly selected customers for training and 1/3 for testing. The classification accuracy achieved was 81%[3].

Methods	Feature selected	Accuracy
Original	16	53%
Encoding 'Job' attribute	21	54%
Using SVM algorithm after feature selection	26	81%

Figure 6 The comparison of results with method from another paper

Feature selection is often a key datamining step, since it is useful to discard irrelevant inputs, leading to simpler data-driven models that are easier to interpret and that tend to provide better predictive performances [4]. In Ref. [5], it is argued that while automatic methods can be useful, the best way is to perform a manual feature selection by using problem domain knowledge, i.e., by having a clear understanding of what the attributes actually mean.

Support vector machines (SVMs) is also one of the classification models. SVM is more flexible when compared it with classical statistical modelling (e.g., llogistic regression) or even decision trees, presenting learning capabilities that range from linear to complex nonlinear mappings [2].

As we can see, using SVM algorithm after feature selection can provide a much better performance than ours. The accuracy of it reaches 81%. The main difference here is because I haven't done feature selections. Business intuitive knowledge was used in this paper to define a set of fourteen questions, which represent certain hypotheses that are tested. Each question (or factor of analysis) is defined in terms of a group of related attributes selected from the original set of 150 features by a bank campaign manager (domain expert). The final set of selected 16 attributes are categorized into 10 factors. By this way, the model can have deeper understanding of each attribute and then provide a much better performance.

In this paper, some attributes like 'job' was eliminated because it's considered to be useless for the prediction. However, I actually slightly improved the performance by encoding the 'job' attribute, which means the 'job' attribute did contribute to predict the right output. So, the feature selection procedure should include a proper combination with theoretical analysis and practical experiments.

5 Conclusion and Future work

5.1 Conclusion

In this paper, it shows how we implement a typical neural network with real world data. We compared the training results from modelling encrypted data and decrypted data, and the comparison shows that the encoding technique improves our model' accuracy a little. It's only a little but be noted that here we only encode 1 out of 16 attributes. During the procedure of building the model, we found that when have 20 hidden units in this two-layer network, the neural network provides the best performance. If we use more than 20 hidden units, their activations will be overlapping, which means the training accuracy won't improve while costing a lot more time.

A comparison with another method used on the same dataset is made. It shows that the neural network will perform much better if we add feature selection during the data pre-processing phase. But it should be noted that the method used by the paper has eliminated a few attributes from the raw dataset. Those attributes are considered meaningless in the classification problem. However, the results from my experiments clearly shows the eliminated 'job' attribute is actually useful to improve the prediction. So, the feature selection technique in this case (the number of attributes in the dataset isn't too many, only 16), might be better if it can keep all the attributes. Elimination is still needed in some other cases when there are too many features and a lot of them are redundant. Overall, all of these methods need future work to be done in order to improve.

5.2 Future Work

We can apply the technique of feature selection to our model. For the list of attributes and factors generated after selection, we use the encoding technique for some of the attributes. Compare the results between adding encoding technique and only feature selection. Try to find out whether the encoding technique can improve the feature selection model furthermore.

Also, the data includes a lot of unknown values in many attributes. Find a way to encode the unknown values is very important in this case. Try to avoid the affection from the value unknown, because predictions might not be reliable when influenced by unknown information.

References

1. Bustos, R. and T. Gedeon. Decrypting Neural Network Data: A GIS Case Study. in Artificial Neural Nets and Genetic Algorithms. 1995. Springer.

2. T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edition Springer-Verlag, NY, USA, 2008.
3. S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014
4. Isabelle Guyon, André Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
5. Ian H. Witten, Eibe Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2nd edition Morgan Kaufmann, 2005.