FIT5186 Intelligent Systems Assignment

# Prediction of potential insurance policy buyers

Ying Xu, 9 September 2012
*Department of Software Engineering*
*Southeast University & Monash University, Suzhou, China*
*yxu132@student.monash.edu.au*

**Abstract** - This research designs and develops a predictor to determine potential insurance buyers according to their private information. The best neural network model is selected from multiple available networks provided by Neural Shell2. Training parameters are optimized by compare the test results of the predictor. Then, several feature selection methods are performed to find the most relevant subset of features to improve the predict accuracy and reduce computing complexity. At last, the contribution of the paper and limitation of current predictor is concluded for further improvement of optimization.

*Keywords* – Neural Network, Insurance Policy, Feature Selection

## I. INTRODUCTION

Insurance companies always do some survey and analyses before they are trying to sell their insurance policy, which can help them focus on the more potential clients. Different people would like to buy different kinds of insurance policies, which is closely related to their personal information. In fact, if we do enough investigation into the insurance policy selling histories, we will find that clients can be divided into different categories and people in the same category would like to buy the same policy. Thus, if we want predict the potential clients for one insurance policy, we can analysis the information of all clients and find the very client set in which clients are more likely to buy the policy than those in others.

In order to solve this problem, a predictor could be trained to make decision for future coming clients. Usually, the training process is based on the history records of insurance buyers and their private information, targeting at finding a pattern for specific category. This pattern can be a linear function which separates different client sets, a decision tree which gives a final judge on the leaf nodes or a multiple layer neural network which models the problem as a complex and high-dimensional decision process. Different learning models have their own strengths and weaknesses, and there exists no such model as perfect for all problems. Decision Tree breaks down a complex decision-making process into a collection of simpler decisions, thus providing a solution which is often easier to interpret [1]. It has relatively low complexity in terms of computing and resource consumption, but is vulnerable with data noise. Linear models have nice analytical properties and form the foundation for more sophisticated models, but it only applicable for linear spreadable problems and have poor performance on problems involving spaces of high dimensionality [2]. Genetic Algorithm is more suited for optimization problems and usually combined with other learning algorithm to achieve better results. Here, we choose Artificial Neural Network to do the predict task as it performs best for high dimensional problems and is robust with noise in the training data. Information of insurance policy buyer is acquired through questionnaire which contains large possibility of noise data. In fact, neural network is quite a strong model as it can model most of the learning problem, except that it is unexplainable. We cannot give the final network a clear interpretation as that in decision tree. No one knows how it comes with the final judgments. However, in order to achieve high accuracy, Artificial Neural Network is used here to do the predict task.

## II. DATA SETS

The data set is provided by an insurance company. The feature of interest is whether or not a customer buys caravan insurance. Per possible customer, 86 attributes are given: 43 socio-demographic variables derived via the customer's ZIP area code, and 43 variables about ownership of other insurance policies.

Some of the continuous value attributes are transformed into several discrete values one to compute more accurate training examples. For example, the age of clients is divided into 6 levels ranging from 20 to 80 years old, replacing the original 60 candidate value by 6 possible selections. One more step could be taken to discrete such multi-value attribute into several binary value attributes. For example,

income range is first replaced by 5 levels, and then transformed to 5 attributes: MINKM30, MINK3045, MINK4575, MINK7512 and MINK123M, which take 0 and 1 as candidate values.
In fact, the data set downloaded is quite well-formatted and the only preprocessing needed here is to save the .txt file into CVS (Comer Split Values) file. So, the only issues we should address before training is choosing the attribute subset which can produce the best predict result. There are 85 attributes which contribute to the final judgment of a potential client. But not all of them are closely related to decide whether one would be interested in the Caravan insurance policy.
Feature Selection is used against learning models which entail large number of measured attributes but with low number of training samples. It can be optimal if an exhaustive search of attribute subsets is performed. But as the high dimension of original attributes, the resulting permutation of them will be considerable. So, a satisfactory set of features will be enough in practice. The final select performance will be affected by two factors: the selection policy and the search policy. Selection policy can be schema-based or schema-independent according to whether it selects features based on the general characteristic of the data or on the final applied machine learning algorithm. If we choose the second, our neural network will be wrapped in the selection process to evaluate selected subsets. However, multiple machine learning algorithms can in return be applied to feature selection. Top level attributes of a decision tree is more likely to be appear in the selected subset; linear support vector machine ranks the attributes based the size of coefficients; instance-based learning decide the importance of an attributes through “near hits” and “near miss” values. All this selection policies will be combined with different search policy, and such combinations will be compared to determine a best subset of attributes.

## III. TRAINING ISSUES

### 3.1 Basic Concepts

The network we used here is multilayer network, as it has been proven to be of the greatest practical value. Neural networks are based on linear regression and they use the function follow the form of (1).

$$y(x,w) = f(\sum_{j=1}^{M} w_j \phi_j(x)). \tag{1}$$

In which $f(.)$ is a nonlinear activation function in the case of classification and is the identity in the case of regression [3]. For neural network, should be extended to allow its dependency on parameters and then we only need to adjust the parameters during training. Figure 1 is a typical two-layer neural network. Information is propagated to through hidden layers and parameters $w_{MD}^{(1)}$ and $w_{KM}^{(2)}$ should be adjust to achieve best result. (2) (3) and (4) shows how the information are propagated and the final output function is given be (5) [3].

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}. \tag{2}$$

$$z_j = h(a_j). \tag{3}$$

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}. \tag{4}$$

$$y_k(x,w) = \sigma(\sum_{j=1}^{M} w_{kj}^{(2)} h(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}) + w_{k0}^{(2)}). \tag{5}$$

Multiple generalization of the network model is available for chosen. One is to absorb the bias parameter w0 into by defining an additional input x0 and set x0=1. Another is to include skip-layer connection. These generalizations allow more general network architecture to model different problems.

### 3.2 Choosing Network

Neural Shell2 provides us with multiple neural networks, including back-propagation networks, probabilistic neural network (PNN), general regression neural network (GRNN) and polynomial net. In standard back-propagation network each layer is connected to its immediate previous layer and errors are back propagated to them. PNN generate probability density function estimate for each category at the output layer and it is very suitable for sparse data. GRNN works by measuring how a far a given record is from the N dimensional space, where N equals to the number of training records. A comparison among all these available neural networks is performed to choose a best network structure

for the target problem, in which training parameters are set to default to gain a general view of each network. The results are show in TABLE 1.

TABLE 1 Comparison between different neural networks

| | | Standard BA(3 layers) | Jump Connection | Jordon net(3 layers) | Ward Net(3 layers) | GRNN(3 layers) | | PNN(3 layers) |
|---|---|---|---|---|---|---|---|---|
| R Square | C1 | 0.0566 | 0.0380 | 0.0517 | 0.0504 | 0.1240 | True positive | 0.8732 |
| | C2 | 0.0565 | 0.0422 | 0.0511 | 0.0491 | 0.1240 | proportion | 0.9464 |
| r Square | C1 | 0.0616 | 0.0479 | 0.0590 | 0.0520 | 0.1796 | False positive | 0.0536 |
| | C2 | 0.0613 | 0.0544 | 0.0571 | 0.0491 | 0.1796 | proportion | 0.1268 |
| Mean Square Error | C1 | 0.053 | 0.054 | 0.053 | 0.053 | 0.049 | | |
| | C2 | 0.053 | 0.054 | 0.053 | 0.053 | 0.049 | | |
| Mean Absolute Error | C1 | 0.100 | 0.088 | 0.094 | 0.100 | 0.103 | | |
| | C2 | 0.100 | 0.086 | 0.095 | 0.107 | 0.103 | | |

By comparison, we can find that GRNN gives the highest R square and lowest Mean Square Error. In fact, other networks including PNN all give Mean Square Error rate around 0.53, and R square below 0.1. So we choose GRNN here as the network to be trained and optimized in the net session.

## IV. RESULTS

4.1 Setting optimization

Number of records in the total set is 5822, and the test set is extracted from these records with a percentage N = 20. Best smoothing factor for the test set is saved and the training process will stop when no further improvement could be done to the current network.

Optimize the initial settings

To train a GRNN, two settings should be specified before training, one is the distance metric, and the other is the Calibration for GRNN. In Neural Shell2, two options are provide for the distance metric: Vanilla or Euclidean and City Block, while three options are given for the calibration for GRNN, they are iterative, genetic and none.

As the genetic option is quite slow given the more than 5 thousands patterns, we choose iterative one as the default calibration for GRNN.

Smoothing factor is a very important parameter in GRNN. Unlike the back-propagation network which rely heavily on learning rate and momentum, GRNN compute a smoothing factor which is applied after the network is trained. Using different smoothing factors when apply the trained network to training set or test set will give different performance. High smoothing rate causes more relaxed surface fits through the data.

Multiple experiments are conducted here to get the best setting for GRNN. The results are shown in TABLE 2.

TABLE 2 Comparison among different settings of GRNN

| Distance Metric | | Vanilla or Euclidean | | | | | | City Block | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| muvfs | | 1.6 | | 3.2 | | 6.4 | | 1.6 | | 3.2 | | 6.4 | |
| Smoothing factor for apply | | 0.58085 | 0.2 | 0.58085 | 0.2 | 0.58049 | 0.2 | 1.599 | 0.2 | 1.87109 | 0.2 | 1.87109 | 0.2 |
| Smoothing factor generated | | 0.5808595 | | 0.5808595 | | 0.580496 | | 1.599609 | | 1.871094 | | 1.871094 | |
| R-s | C1 | 0.12 | 0.61 | 0.12 | 0.61 | 0.12 | 0.61 | 0.15 | 0.62 | 0.09 | 0.61 | 0.09 | 0.61 |
| | C2 | 0.12 | 0.61 | 0.12 | 0.61 | 0.12 | 0.61 | 0.15 | 0.62 | 0.09 | 0.61 | 0.09 | 0.61 |
| r-s | C1 | 0.18 | 0.62 | 0.18 | 0.62 | 0.18 | 0.62 | 0.27 | 0.62 | 0.19 | 0.62 | 0.19 | 0.62 |
| | C2 | 0.18 | 0.62 | 0.18 | 0.62 | 0.18 | 0.62 | 0.27 | 0.62 | 0.19 | 0.62 | 0.19 | 0.62 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mean Square Error | C1 | 0.049 | 0.022 | 0.049 | 0.022 | 0.049 | 0.022 | 0.048 | 0.022 | 0.051 | 0.022 | 0.051 | 0.22 |
| | C2 | 0.049 | 0.022 | 0.049 | 0.022 | 0.049 | 0.022 | 0.048 | 0.022 | 0.051 | 0.022 | 0.051 | 0.22 |
| Mean Absolute Error | C1 | 0.103 | 0.035 | 0.103 | 0.035 | 0.103 | 0.035 | 0.101 | 0.032 | 0.105 | 0.032 | 0.105 | 0.32 |
| | C2 | 0.103 | 0.035 | 0.103 | 0.035 | 0.103 | 0.035 | 0.101 | 0.032 | 0.105 | 0.032 | 0.105 | 0.32 |

From this table, we can find that, when distance metric is City Block, and the maximum upper value for search is 1.6, GRNN give the best results. When apply the resulting network to the whole pattern file with smoothing factor 1.599609, the R square is 0.15 and the accuracy is 95.2%, which is 0.1% higher than other networks. Although all the networks having City Block as distance metric give the same accuracy at 97.8% when smoothing factor is 0.2, the network who set maximum upper value for search as 1.6 gives the best R square at 0.62. So, we choose GRNN whose distance metric is City Block and maximum upper value for search is 1.6 as the final network for training.
Up to now, we get the best network model and best settings for this model. We will use this neural network to find the most relevant feature subset in the next session.

### 4.2 Feature selection

Weka [4] provides 17 evaluators and 11 search methods for feature selection process. Here we only test several most broadly used combinations to do experiments and comparison. Results are shown in TABLE 3.

TABLE 3 Comparison among different feature selection methods

| | CfsSubsetEval+BestSearch | CfsSubsetEval+GeneticSearch | ChiSquaredAttributeEval+Ranker | GainRatioAttributeEval+Ranker | ConsistencySubsetEval+GeneticSearch |
|---|---|---|---|---|---|
| Num. of features selected | 8 | 36 | 39 | 37 | 45 |
| Mean Square Error(sf=0.2 C1) | 0.052 | 0.021 | 0.022 | 0.042 | 0.020 |

Notes: CfsSubSetEval evaluates predictive value of each attribute individually, along with the redundancy within them; ChiSquareAttributeEval computes the chi-squared statistic of each attribute with respect to the class; GainRatioAttributeEval evaluates attribute based on gain ratio; ConsistencySubsetEval projects training set onto attribute set and measure consistency in class values.

From this table, we find that ConsistencySubsetEval and GeneticSearch give the best result. It selects out 45 features for training work and improves the final accuracy to 98%. With such a compacted training set, training time is reduced considerably from 05:03 to 02:51, which is almost half of the origin, while the accuracy is 0.2% higher than before.

## V. LIMITATIONS

Neural Network is very strong at modeling complex and noisy problems. However, its computation complexity is quite high due to its large amount of hidden neurons and propagation process. A two-category neural network with 10 input neurons and 10 hidden neurons will result 120 weights, which will be accessed at least once for each incoming pattern, and this is just the case for feed-forward neural network, let long the back-propagation one. On the other hand, how to choose the best structure and number of hidden neurons of the network is another difficult problem. If we model this problem as a searching process, the computation cost will be too high to leverage the benefit we gain from the network. So, we can never find the best but the most suitable model for our specific problem.
Although Neural Shell2 and Weka provide very strong design structure for neural network and machine learning designers, they are limited by their original inner settings. We cannot use the model as dynamic as possible. For example, the GRNN in Neural Shell2 only give the 3-layer structure,

however, better results will be generated by more complex structures. But we can check that in this research.

## VI. Conclusion

In this research paper, a neural network is designed to predict the potential clients of CARAVAN insurance policy and the accuracy is as high as 98%. Multiple experiments and comparisons are conducted to optimize the whole design. At first, by compare different neural network, we find that GRNN is the most suitable one for this problem. Secondly, network specific settings are tested to decide a best state for the following feature selection experiments. Finally, several technologies are used to search and select a feature subset which contributes most to the final judgment of a potential insurance buyer. After such an optimization process, we not only find the most suitable predictor for this particular problem but also find out a more efficient and compact subset of features to improve the performance of our predictor.

## References


[1] Safavian, S.R. and D. Landgrebe, *A survey of decision tree classifier methodology.* Systems, Man and Cybernetics, IEEE Transactions on, 1991. **21**(3): pp. 660-674.

[2] Bishop, C.M. and SpringerLink, *Pattern recognition and machine learning*. 2006: Springer New York, p. 138.

[3] Bishop, C.M. and SpringerLink, *Pattern recognition and machine learning.* 2006: Springer New York, pp. 225-232.

[4] Witten, I.H., E. Frank, and M.A. Hall, *Data Mining: Practical machine learning tools and techniques*, 2011: Morgan Kaufmann, pp. 487-494.