Learning Environment for Life Time Value Calculation of Customers in Insurance Domain

Andrea Tettamanzi[1], Luca Sammartino[2], Mikhail Simonov[2], Massimo Soroldoni[2], and Mauro Beretta[3]

[1] Università degli Studi di Milano, Dipartimento di Tecnologie dell'Informazione
Via Bramante 65, I-26013 Crema (CR), Italy, andrea.tettamanzi@unimi.it
[2] Nomos Sistema S.p.A., Viale Monza 259, I-20126 Milano (MI), Italy,
{simonov, soroldoni, sammartino}@nomos.it
[3] Genetica S.r.l., Via San Dionigi 15, I-20139 Milano (MI); Italy
beretta@genetica-soft.com

Abstract. A critical success factor in Insurance business is its ability to use information sources and contained knowledge in the most effective way. Its profitability is obtained through the Technical management plus Financial management of the funds gathered on the market. The profitability of a given customer can be evaluated through its Life Time Value (LTV). We aim at applying evolutionary algorithms to the problem of forecasting the future LTV in the Insurance Business. The Framework developed within the Eureka co-funded research projects HPPC/SEA and IKF has been adapted to the Insurance Domain through a dedicated Genetic Engine. The solution uses RDF and XML-compliant standard. The idea of using evolutionary algorithms to design fuzzy systems date from the beginning of the Nineties and a fair body of work has been carried out throughout the past decade. The approach we followed uses an evolutionary algorithm to evolve fuzzy classifiers of the data set.

1 Introduction

A critical success factor for an Insurance business today is its ability to use information sources and contained knowledge in the most effective way. This strategic use of data can result from opportunities presented by discovering hidden, previously undetected, and frequently extremely valuable facts about consumers, retailers and suppliers, business trends, and direction and significant factors. Knowing this information, an organisation can formulate effective business, marketing, and sales strategies; precisely target promotional activity; discover and penetrate new markets; and successfully compete in the marketplace from a position of informed strength.

Ontologies can play an important role in industrial systems by enabling access to legacy resources in a transparent and distributed way. Since ontologies are developed in order to provide a machine-processable semantics of information that is exchanged between different agents, either humans or software, the same mechanism can be used

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 1251–1262, 2004.

for Legacy Service discovery and for automated reasoning about the content of the answers obtained from such a system. However, the only possibility to offer the self-learning environment and DSS system is the use of data mining. The technology aimed at achieving this strategic advantage is known as 'advanced data mining'.
The Framework developed within the Eureka co-funded research projects HPPC/SEA and IKF has been adapted to the Insurance Domain through a dedicated Genetic Engine. The solution uses an RDF- and XML-compliant standard according to W3C recommendations for data representation.
One interesting application of data mining is customer modeling, whose practical uses in Finance range from proactive marketing to fraud detection. We will focus on a particular real-world application of the above methods within the Insurance Domain, namely the calculation of the Life Time Value of customers.
This document is organised as follows: Section 2 introduces the problem and its formulation; Section 3 provides a sketch of the system; Sections 4 and 5 present a descriptions of adopted business and fuzzy modeling; Section 6 illustrates an optimization engine based on evolutionary algorithm for the problem; Section 7 provides a sample application of the proposed approach and the results obtained; finally, Section 8 outlines at a glance the main achievements of the project.

2 Calculating the Customer Lifetime Value

Profitability in the Insurance Business is obtained through the Technical management plus Financial management of the funds gathered on the market. Since interest rates have dropped dramatically in the recent past, current financial management cannot anymore guarantee the profitability of the whole business by itself. As a consequence, the only way to run such a business properly is to increase the profitability of the Technical Management of insurance policies. The Insurance Business is customer-dependent and it is obvious that there are "good" and "bad" customers in terms of profitability. The typical portfolio of an insurance company contains both of them. The profitability of a given customer can be evaluated through its Life Time Value (LTV). While the past LTV is known and the present LTV can be calculated on the fly, the expected future LTV can only be forecasted. We illustrate an application of evolutionary algorithms to the problem of obtaining an accurate and reliable forecast of the future customer LTV.
The insurance business still relies heavily on legacy applications, where several millions of customers plus several insurance policies for each of them are managed. The huge amount of data makes it impossible to manage an individual customer in a personalized fashion. This is a typical business problem in the insurance domain which can be successfully solved by applying advanced data mining techniques.

3 Overview of the System

All back-office management activities in the Insurance domain are performed by huge and well-tested legacy systems. Through an intranet, an insurance agent deals with a legacy system in order to obtain needed information or to answer typical queries from

customers that visit the agency or call it over the phone. The agent deals with the back office when more complex questions are raised or real-time calculations are required. This process is time-consuming and automation through semantic capabilities and data mining software would be helpful. Unfortunately, software with such a mix of automated reasoning and advanced data mining with self learning capabilities is not available in commercial systems.

On the other hand, users can deal with web-based applications during the purchase of an insurance product or when the "automated" customer relationship management (CRM) is contacted for frequently asked questions. Managing such relations is not straightforward, because it requires accessing a legacy system, understanding user queries, knowledge of insurance regulations and guidelines with user data or even profiles, etc. By providing such capabilities, also known as customer services, the insurer invests money and expects to increase the profitability of the main business. Accordingly, it is very important to be able to predict the Life Time Value of a customer in order to ensure the best possible customer service with respect to the expected profit.

Industrial applications must process some specific calculations in order to come up with a customer model and apply it to specific customer data. In order to allow these calculations, some back-office activities must be performed and especially the model must be created, updated, and adapted to new realities emerging from a continuously changing customer portfolio.

The system is implemented as a client-server solution, made up of a Genetic Engine and a Client controlling it. The communication mechanism is based on raw sockets in order to be fully portable between different platforms (Windows, Unix, Linux etc.), and can be deployed on parallel machines (MPI libraries are used). The enterprise solution is implemented in C++ and the communication between client and server is done in XML according to W3C recommendations. A wrapping layer is implemented in order to allow the call of the whole system from the Java environment, such as a distributed web application. The IKF vertical applications[1] use this interface to perform knowledge extraction from structured data sources. A specific HTML (JSP) application was also implemented to allow the front office to call the application to evaluate a model for a given customer and obtain a prediction of the future LTV.

4 Business Modelling

In the Insurance Domain the central point of any modelling is the customer. Accordingly, the Customer is modelled as the owner of relationships with the Insurer and other people. Furthermore, the relation between the Customer and his/her Properties is modelled, since Risks are generated by Properties. Risks can be the object of the Insurance Contract and the latter can generate profits or losses.

The specific Property named Life can be also modelled in order to expand the system towards Life Insurance contracts and investments in general (we observe that investments do not imply insurance-relevant life risks since the only risk in investment is financial).

[1] see www.ikfproject.com for further details.

People become customers because they own some properties. Properties generate risks and people wish to be sure that losses can be repaid in order to have the possibility to reconstruct their patrimony, i.e. people have Markov's needs. Markov's needs are considered goods to be sold on the Insurance Market. But the owners of Markov needs are not all equal: there are people who carry more risks and people who carry fewer risks, people who will generate profits, no profit, or losses. Moreover, there are people who generate losses in the short period but generate a high profit over the life time relation. Having said this, it is very important to be able to calculate the Life Time Value of a customer (LTV) for the past and present, and to be able to predict the LTV for the short- and medium-term future, as well as for the entire residual lifetime.

The application offers (Fig. 1):

- an automated model development as a support for a back office activity
- a concrete system for customer classification in order to define segments with customers having high expected LTV, medium LTV and low LTV able to generate in future high profit, medium profit and low profit respectively
- a "teller machine" able to apply the developed model to users data, useable even in a distributed web environment
- an advanced data mining genetic engine able to perform data discovery relevant for a given business
- XML/RDF compliant results generation accordingly W3C recommendations able to deliver not only the results but the semantic meaning of results

Setting intuition to the minimum, an Insurance business could be modelled according to the following schema[2]:

- Properties –implies– Risks (1..*)
- Life (is-a Property) –implies– Risks (1..*)
- Customer –has– Properties (1..*)
- Customer –has– Policy (1..*)
- Policy –has– Claims (0..*)
- Policy –has– Receipts (1..*)
- Claim –has– Payments (1..*)
- Customer - has - Relationship with the Insurer (1..1)

The schema means that a customer has at least one policy, for which at least one receipt is expected (incoming cash flow for Insurer), and at least zero claims can be expected. A claim expects at least one payment (cash outflow). Following such schema, user data are represented in a physical legacy database (DB2 is widely adopted in the use case).

Our application relies on the legacy database, but it also provides other ways to access insurance-related information, and combines legacy data with that information by integrating IES domain ontology with the legacy db.

The domain modelling was performed as follows:

Object	DOLCE-Lite + Type	Informal description	Informal constraints

[2] The above mentioned modelling comes from the IKF-IES application developed within Eureka co-funded project named IKF, see **http://www.ikfproject.com**. IKF develops frameworks to allow knowledge management in various business settings.

Policy	Non-physical object	A policy is issued by an insurer for an insured and covers certain losses given certain risks	Held_by Insured Issued_by Insurer Covers Insured
Insurer	Non-physical object	An insurer issues a policy to an insured which covers losses given certain risks	Issues policy Pays or disclaims claims
Insured	Non-physical object	An insured is covered by a policy against certain losses given certain risks	Covered by a policy
Claim	Activity	An insured makes a claim against a policy for a loss	Made by insured Paid or disclaimed by Insurer
Risk	Non-physical object	Risk is the likelihood of a certain loss for a certain insured	Contemplated by a policy
Loss	Non-physical object	A loss exists when an insured suffers some injury to person or property. The loss is said to be covered if it was part of the risk contemplated by the policy	Suffered by insured Covered (or not) by a policy
Disclaimer	Activity	A disclaimer is made by an insurer when a loss is not covered by a policy, usually when not a contemplated risk	Made by insurer

Fig. 1. A table used for experts' drafting of the core ontology

In order to develop the model, a task force of domain experts, software engineers, and ontology specialists has reached a design agreement. The model was developed and the genetic engine was specially engineered to match the domain.
The Management Framework is responsible for generating a model (back-office activity), evolving a model evolution to keep it updated (back-office activity), and evaluating a customer (front-office activity) during the policy subscription or call center contacts (even during self help use through web channels).

5 Fuzzy Rules Model

The task of generating predictive models of customer behaviour, or classifiers, can be (and has been) approached in many ways and using many techniques, including decision trees, neural networks, linear classifiers and non-linear statistical classifiers, among the most popular ones. All techniques have their advantages and drawbacks. However, one thing that is often required in practical applications is the interpretability, or explanatory power, of a model.
This is the main motivation for evolving classifiers made of decision rules. Moreover, because reality is not sharp and clean and in general decisions taken using crisp

thresholds are dangerous in a number of real-world applications, we have moved our attention to fuzzy rules, for their intrinsic interpolative behaviour.
The idea of using evolutionary algorithms to design fuzzy systems date from the beginning of the Nineties [5, 12] and a fair body of work has been carried out throughout the past decade [6, 7, 4]. The approach we followed is largely based on the development of previous work on the evolution of fuzzy controllers [11], which has been already validated on standard machine learning benchmarks [10].
Based on that work, we define a classifier as a rule base, of up to 256 rules, which should be more than enough to express even very complicated models, each rule comprising up to four antecedents and one consequent clause. Up to 256 input variables and one output variable can be handled, described by up to 16 distinct membership functions each. Membership functions for input variables are trapezoidal, while membership functions for the output variables are triangular.

6 Genetic Engine

Our approach uses an evolutionary algorithm to evolve fuzzy classifiers of the data set. Evolutionary algorithms are a broad class of optimisation methods inspired by Biology, which build on the key concept of Darwinian evolution. It is assumed that the reader is already familiar with the main concepts and issues relevant to evolutionary algorithms; good reference books are [1, 3, 8, 2].

6.1 The Algorithm

An island-based distributed evolutionary algorithm is used to evolve classifiers. An island-based algorithm maintains several populations (islands) which evolve separately exchanging individuals from time to time. The sequential algorithm executed on every island is a standard generational replacement, elitist evolutionary algorithm. Crossover and mutation are never applied to the best individual in the population.

Encoding
Classifiers are encoded in three main blocks: a set of membership functions for the input variables, a set of symmetric membership functions, represented by (area, center of mass) pairs for the output (or classification) variable, and a set of rules.
A single input variable membership function is defined by four fixed-point numbers, each fitting into a byte. Output variable membership functions are assumed to be symmetrical, and thus can be described using just two parameters: their area and center of mass. A rule is a list of up to four conjoint antecedent clauses (the IF part) and a consequent clause (the THEN part). A clause is represented by a pair of indices referring respectively to a variable and to one of its linguistic values, i.e., a membership function.

Initialization

The population can be seeded either with hand-written or otherwise already existing classifiers or with new random ones. A new random individual is created according to the following algorithm:

Each input variable has at least one linguistic value; the number of additional values is determined by sampling from a truncated exponential distribution with mean three;

The shapes of the membership functions for the input variables are determined by randomly extracting a center C with uniform probability over the variable definition interval, and a spread σ from an exponential distribution with mean *1/4* of the variable range. The four numbers defining the trapezoid, a, b, c and d are assigned as follows:

$$\begin{aligned} a &= C - 2/3\sigma, \\ b &= C - 1/3\sigma, \\ c &= C + 1/3\sigma, \\ d &= C + 2/3\sigma \end{aligned} \qquad (1)$$

The exponential distribution is used to determine the spread because it is the zero-information probability distribution for a non-negative random variable. At least two output membership functions have to be present for each output variable; the number of additional linguistic values for each output variable is determined by sampling from a truncated exponential distribution with mean three. The centers of mass for the output variable are randomly extracted in the [0, M-1] interval; the areas are extracted at random such that they correspond to a triangular membership function whose base is entirely contained in that range. At least M rules have to be present, using at least M different output linguistic values; the number of additional rules in the rule base is determined by sampling from a truncated exponential distribution with mean six. The rules are generated according to the following algorithm:

for each input variable, a fair coin is flipped to decide whether to include it in the antecedent part, not exceeding four variables;

for each selected input variable, a linguistic value is extracted among those defined;

an output variable and its linguistic value are extracted for the consequent part of the rule.

Recombination

The recombination operator is designed to preserve the syntactic legality of classifiers. A new classifier is obtained by combining the pieces of two parent classifiers. Each rule of the offspring classifier can be inherited from one of the parent programs with probability *1/2*. When inherited, a rule takes with it to the offspring classifier all the referred linguistic values with their membership functions. Other linguistic values can be inherited from the parents, even if they are not used in the rule set of the child classifier, to increase the size of the offspring so that their size is roughly the average of its parents' sizes. This sort of recessive genes might fall back into use because of mutations, and it has been empirically observed that their presence enhances the performance of the algorithm.

Mutation

Like recombination, mutation also produces only legal classifiers. Mutation can result in one or more of the following changes, with probability given by the mutation rate, p_m, identical and independent for each component of the genotype:

- a new linguistic value with a random membership function is added to an input variable;
- a linguistic value whose membership function is not used in the rules is removed from an input variable;
- a membership function is perturbed as follows: each of the four points defining the trapezoid can be moved, with probability p_m, to a new random position between the previous and next points;
- a new linguistic value, with random area and center of mass, is added to an output variable;
- a linguistic value whose area and center of mass are not used in the rules is removed from an output variable;
- an area-center of mass pair is perturbed as follows:
  1. a standard deviation σ for the perturbation is extracted from an exponential distribution;
  2. a new center of mass is extracted from a truncated normal distribution with mean the old center of mass and standard deviation σ;
  3. a new area is extracted from a truncated exponential distribution with mean the old area, such that it corresponds to a triangular membership function entirely contained in the range of the relevant output variable;
- a new random rule is added to the rule set; the new rule is generated as follows:
  1. for each input variable, a fair coin is flipped to decide whether to include it in the antecedent part, not exceeding four variables;
  2. for each selected input variable, a linguistic value is extracted among those defined;
  3. an output variable and its linguistic value are extracted for the consequent part of the rule;
- a rule is removed from the rule set;
- a rule gets a random antecedent clause predicating an input variable not yet used added to it;
- an antecedent clause is removed from a rule;
- the predicate of an antecedent clause is modified by randomly extracting one of the linguistic values defined for the relevant input variable;
- the predicate of the consequent clause of a rule is modified by randomly extracting one of the linguistic values defined for the relevant output variable.

Migration

Migration is responsible for the diffusion of genetic material between populations residing on different islands. At each generation, with a small probability (the migration rate), a copy of the best individual of an island is sent to all connected islands and as many of the worst individuals as the number of connected islands are replaced with an equal number of immigrants.

Moreover, an immigrant whose fitness is lower than the lowest fitness in the island population is always discarded.

6.2 Fitness

The fitness function is basically a logarithmic likelihood function, which measures how well the predictions made by the model match with the dataset used for learning. A particular care has been taken to assign a relative weight to false-positive and false-negative predictions, to reflect the fact that the loss incurred as a consequence of mistaking a bad customer for a good one are in general a multiple of the loss caused by missing a good prospective customer.

6.3 Selection

Elitist linear ranking selection, with an adjustable selection pressure, is responsible for improvements from one generation to the next. Overall, the algorithm is elitist, in the sense that the best individual in the population is always passed on unchanged to the next generation, without undergoing crossover or mutation.

7 Case Study and Experimental Results

The system has been tested in a real world application in order to validate its results; in order to complete this task, a real dataset coming from an insurance company has been collected, and a special hardware and software configuration has been set up. In the next paragraph we illustrate the experimental results achieved.

Case Study
As mentioned above, a dataset of customers obtained from an insurance company has been collected with the objective of predicting customer values to minimize risk for the company. That is, first of all we collected insurance company database containing relevant customer data required for a comprehensive analysis of each customer profile (historical personal data); starting from there, we extracted from the database only those fields considered relevant to our analysis, obtaining a dataset of 725,035 records (customers), containing 20 fields each.
A rating (*a posteriori* evaluation of each customer expressed by a number in the [0, 1] interval, where 0 is worst and 1 is best) has been added as the last row of this dataset; such value has been computed using the following equation:
Normalized Score = (Historical customer value + Upper bound of Historical customer value) / (Upper bound of Historical customer value + Lower bound of Historical customer value).
Once this task has been completed, we have analyzed the resulting dataset in order to examine the distribution of the rating; the results of our analysis can be described by the following tables:

Dataset analysis - observed value by range				
min	max	min	max	occurrency
- 10.000	- 8.400	-	0,1	4.160
- 8.400	- 6.800	0,10	0,2	706
- 6.800	- 5.200	0,20	0,3	1.055
- 5.200	- 3.600	0,30	0,4	1.864
- 3.600	- 2.000	0,40	0,5	3.857
- 2.000	- 400	0,50	0,6	14.694
- 400	1.200	0,60	0,7	661.309
1.200	2.800	0,70	0,8	29.067
2.800	4.400	0,80	0,9	6.493
4.400	6.000	0,90	1,0	3.655

Rating distribution	
hyst. value	scoring
-5000	0,3125
-4000	0,3750
-3000	0,4375
-2000	0,5000
-1000	0,5625
-500	0,5938
-200	0,6125
-100	0,6188
-50	0,6219
-10	0,6244
-5	0,6247
0	0,6250
5	0,6253
10	0,6256
50	0,6281
100	0,6313
200	0,6375
500	0,6563
1000	0,6875
2000	0,7500
3000	0,8125
4000	0,8750
5000	0,9375

Fig. 2. Dataset analysis - observed value by range

A first consideration has been made: the occurrence of customers with a value within the [0.6-0.7] interval make our dataset strongly unbalanced, and this situation could compromise the reliability of our predictions. Therefore, our next step has consisted in sampling the records to obtained a more balanced dataset; once this operation has been completed, records within the [0.6-0.7] interval have been reduced to 46,291 (instead of 661,309), giving us a final dataset composed by 110,016 total records.
At this point, a classical machine learning protocol has been adopted by splitting the dataset in three different subsets; first of all, a **training** dataset of 20% records has been extracted in order to calculate individual fitness. Then, it has been extracted a test set of 70% records to which each single generation best model calculated by evolutionary algorithms has been applied, in order to obtain a 'test fitness value'.
As long as this test fitness increases, the evolution is carried on; as the test fitness begins to decrease (due to overfitting), evolution has been automatically stopped and the last best model saved. Finally, a **validation** dataset of 10% of records has been extracted in order to validate the whole approach by applying it to the best model previously produced with a specific goal: to validate the resulting best model with customer data never used either for training or test.

Experimental Setting

Once the dataset has been prepared, the physical environment to perform session tests has been set up; eight bi-processors server equipped with Pentium IV 2.4GHz and 1 GB RAM running under Windows 2000 Server communicating by fast Ethernet TCP-IP protocol has been used as evolutionary algorithm server stations.
Here we provide the details of the evolutionary software environment adopted:

- Dataset name: Reale.xml (consisting of 110,016 records: 20% used for training, 70% for test and 10% for validation purposes);

- Initial population: random;
- Number of islands:16 (2 for each server station);
- Island population size: 50 individuals;
- Crossover rate: 60%; Mutation rate: 15%; Migration rate: 1%;
- Linear ranking selection pressure: 1.8; Environment islands topology: ring.

Results

The evolutionary algorithm automatically stopped its computation when 1,232 generations had been completed on Server #4 (this is the highest generation of the whole environment) due to satisfaction of the self-termination condition with a total computation time of 22 hours and 46 minutes approximately.
The results reached are listed below:

Dataset	Average fitness	Standard deviation
Training	0.44	0.018
Test	0.39	0.023
Validation	0.37	0.027

In order to complete exhaustively our experimentation, a further statistical analysis has been performed on the incidence of fields in the best model (in terms of their weights inside the best model fuzzy rules); the following table shows the main fields report:

Field name	Incidence on model
Number of claims	48.48%
Policy type code	15.12%
Age of customer	12.21%
Duration of relationship w/ customer	9.01%
Profession	6.24%
County of abode	6.03%
Number of COI instalments	3.02%

A brief consideration on these results is in order: Even if the field with greatest incidence could be easily predictable (Number of claims), the other results are quite interesting (especially the second place of Policy type code) and has been considered extremely interesting by the insurance company, because they give (combined with a deep analysis of best model fuzzy rules) a powerful instrument for customer value calculation.

8 Conclusions

It is important to be able to calculate correctly the expected future life time long value of LTV in order to run a sustainable business and to retain customer fidelity for a long period of time. Even by calculating the future LTV and by selecting the only profitable (during short time) customers, the above mentioned customers may result in losses in the long run. Our experiments clearly showed that the availability of advanced data mining tools in the insurance domain is the only way to assist the Technical Management in their effort to obtain maximum profitability.

Acknowledgments. Some of the achievements described in the article are the result of participation in Eureka co-funded projects HPPC/SEA and IKF. The work described in this paper draws upon the contribution of many people, to whom the authors are indebted. Of course the authors are the sole responsible of any possible mistake. The authors would like to thank the IKF Consortium for their support.

References

[1] T. Bäck. Evolutionary algorithms in theory and practice. Oxford University Press, Oxford, 1996.
[2] C. Darwin. On the Origin of Species by Means of Natural Selection. John Murray, 1859.
[3] L. Davis. Handbook of Genetic Algorithms. VNR Computer Library. Van Nostrand Reinhold, New York, 1991.
[4] R. Dawkins. The blind Watchmaker. Norton, 1987.
[5] L. J. Fogel, A. J. Owens, and M. J. Walsh. Artificial Intelligence through Simulated Evolution. John Wiley & Sons, New York, 1966.
[6] D. E. Goldberg. Genetic Algorithms in Search, Optimization & Machine Learning. Addison-Wesley, Reading, MA, 1989.
[7] J. H. Holland. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, 1975.
[8] J. R. Koza. Genetic Programming: on the programming of computers by means of natural selection. The MIT Press, Cambridge, Massachussets, 1993.
[9] Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Berlin, 1992.
[10] M. Beretta, A. Tettamanzi. Learning Fuzzy Classifiers with Evolutionary Algorithms. Proceedings of the 4th Italian Workshop on Fuzzy Logic (WILF 2001), Physica Verlag, 2002.
[11] A. Tettamanzi. An Evolutionary Algorithm for Fuzzy Controller Synthesis and Optimization. IEEE International Conference on Systems, Man and Cybernetics, Vancouver, Canada, 1995.
[12] A. Berson, Stephen J. Smith, Data Warehousing, Data Mining & OLAP, McGraw Hill; New York, 1997.