Design of an Application for Credit Scoring and Client Suggestion

Fábio Silva, Cesar Analide

Abstract — Risk assessment on loan application is vital for many financial institutions. Most financial institutions have already applied methods of credit scoring and risk assessment in order to evaluate their clients in terms. These systems are often based on deterministic or statistical algorithms. In this context, techniques from artificial intelligence and data mining present themselves as valid alternatives to build such classification systems. In this paper some studies are conducted to evaluate the effectiveness of neural networks as a classification system and improvements upon those classifiers are proposed. Furthermore, a suggestion algorithm is also presented to help clients whose loan applications are refused and provide some explanation on why their loan is refused. Finally an agent based architecture is presented to integrate all algorithms presented in this paper.

Index Terms — Artificial Intelligence, Behaviour Prediction, Credit Scoring, Data Mining, Risk Assessment.

1 Introduction

Nowadays people are becoming increasingly dependent in loans from financial institutions. However it is not an uncommon situation the fact that some people are incapable of correctly assessing the type and amount of a loan that is in fact affordable to them. As a consequence some people tend to delay their monthly instalments or in extreme cases they even become incapable of repaying their debt back to the financial institution. In this context many of financial institution are implementing or improving client classification systems in order to select good from potential bad clients.
Statisticalanalysis and deterministic systems are still the most common classification systems financial institution use in their applications, and there is here a practical opportunity to develop alternative systems based on artificial intelligence [1]. In fact, artificial intelligence and data mining poses an interesting appeal when considering that they may help making semi-autonomous or even completelyautonomous decision mechanisms capable of being updated and react to changes and new trends in an almost real time. Only in present times are these financial instructions conducting studies in order to evaluate how techniques from artificial intelligence and data mining can be used to predict client behaviour [2],[3]. This paper is particularly aimed at credit scoring systems using previous records from old clients to predict and avoid those classified as bad client in terms of debt repayment.
This paper starts by this introduction followed by a section where related work is discussedand present relevant classification models used by financial institutions today are presented. In this section it is also discussed some approaches made seen in the literature to improve classification algorithms. This section ends with a reference of present toolkits and frameworks available today to use when developing classification systems.
In section 3 the dataset used in this paper is presented. The next section, section 4 reveals a study conducted to improve the performance of neural networks in the problem at hand. In the end of this section results from the algorithms proposed are shown and performance increases observed.
Later in section 5 a case study to support the implementation of suggestive algorithms in this context is presented as well as a suggestive algorithm based on genetic programming and the improved classifier explained in section 4.
In section 6 an agent based system architecture is proposed to integrate all algorithms developed and provide all those functionalities in a real world application prototype.
Finally this paper ends with conclusions of the work developed and some indication of future work to further develop these mechanisms.

2 Related work

An evaluation of the state of the art in risk assessment and decision models implemented in some financial institution is presented. It is discussed models which were implemented and even some models in study for client classification. Furthermore, a review of some algorithms and optimizations upon

them is also reviewed as well as some tools and frameworks currently available to use today that may help build automatons decisions systems.

2.1 Models

Some particularities exist when building decision models that must be taken into consideration. Legal issues for example must be of prime concern as any decision made that does not met the appropriate laws are considered illegal and must be disregarded. Client discrimination based in attributes such race orgender is generally illegal in most countries and may justify legal suites those who ignore these considerations. Consequently these models should take into consideration all the aspects stated above.
Different models and approaches can be found in the literature when dealing with credit scoring and risk assessment.
Most financial institutions use statistical pattern recognition models to build their own decision mechanism. In Czech and Slovak Republics' financial institutions the most used technique is the *Logit Analysis* which is an improvement upon the *Linear Discriminant analysis* technique [1].
In a Jordanian bank studies were conducted to evaluate the benefit of using M*ulti-Layer Feed Forward Neural Networks* [2]. Their study led to the conclusion that these structures are in fact good classifiers achieving up to 95% correct evaluations in their tests.
Improvements on standard neural networks classifications using genetic algorithms were also proposed. In this case genetic algorithms are used to optimize the weight calculation in neural networks [3].
Another classification models found use card based reasoning. These systems use data from past events characterized by a set of attributes. Similarities between past cases and present cases are calculated using an appropriated function and the final classification is made based on the most similar case [4].
Different approaches make use of financial liquidity to forecast a client's ability to pay a future instalment. From an historical set of financial liquidity of clients and their behaviour and by comparison with a present client its risk is calculated and appropriate action may be taken before transgression happens if necessary [5].

2.2 Algorithms

As noted above, artificial intelligence techniques use machine learning techniques and data mining to produce results. In this aspect it can be found in the literature several algorithm optimisation proposals.
Improvements in genetic algorithms used for classification exist and can be found in *The Two Stage Genetic Programming Algorithm*. This algorithm produces a set of if-then rules as well as a function based in genetic programming to classify instances [6]. Another example uses a combination of decision trees with genetic programming and is also able to improve classification [7].
Neural networks are also focus of optimisation and some approaches try to make use of feature selection algorithms before constructing the neural network, making some attributes more relevant in this structure [8]. Feature selection using decision trees may be also used to determine a set of attributes, those in the upper levels of the tree, to build the subset of attributes that is considered to be used with the *Naïve Bayes* classifier [9].
All these algorithm combinations obtain improved results when compared to those versions where they are not combined leading to the conclusion that combining different algorithms maybe a good source of optimization.

2.3 Tools and Frameworks

To build any credit evaluation application many financial institutions make use of existing frameworks with a large set of techniques to help the process of data mining.
In this context open source tools like *RapidMiner*[10] or *Weka*[11] provide a vast list of data mining and machine learning techniques that can be used in conjunction with any other application. Those two tools also provide libraries that can be imported to custom programs. *Rapidminer*[10] for instance, as stated in their webpage, is used by the Bank of America. In fact both of the above tools are referenced in many credit evaluation papers.
In a more specialized context, namely neural networks, it can be found*Encong*[12], a comprehensive framework for neural networks.
Commercial tools also exist and provide the same characteristics. In this field we can name *NeuralSolution*[13] as a program that provides a complete framework for neural networks usage.
In this paper the *Weka Toolkit*[11]was used to perform the tests upon some of the algorithms proposed. This decision was made due to the fact that this framework has a collection of machine learning algorithms for data mining

tasks whose can be applied directly in a dataset or used in a java program. *Weka* [11]has also an active support community and their program is released as open source software.

3 PROBLEM DESCRIPTION

In this paper the problem of client classification for loan applications will be considered. The objective is to improve classification modelsbased on neural networks and build a system that is able to dynamically update itself as new data becomes available in an autonomous way.
In order to achieve better results in client classification some past data about a set of attributes characterizing each client of a financial institution is taken into consideration and is used to build the classification model.
Furthermore the system must also be able suggest clients explanation on how they can improve their situation increasing their chances to have loan applications accepted by the financial institution.
The goal of this algorithm is to indicate a client what are the most advantageous characteristics he may possess to be granted with a loan application. With an incomplete set of attributes characterising the client the system will be able to find with the help of its own classification models the set of missing attributes the client must possess. This suggestion algorithm may also be used to promote new services or financial products to such clients.

4 THE DATASET

In this work a dataset related to credit scoring was chosen from the UCI repository [14]. The choice fell upon a German credit dataset, where each client is characterised by a set of 20 attributes, followed by the classification of each customer. This dataset has two versions: one which contains categorical/symbolic attributes and another where these attributes were all transformed into numerical attributes. For simplicity and opportunity, the version which was chosen was the one with numerical attributes.
The dataset itself is a combination of personal, social and financial information about past bank clients. The complete list of attributes is shown in Table 1.

TABLE 1
LIST OF ATTRIBUTES OF THEGERMAN CREDIT DATASET.

Number	Attribute
1	Status
2	Duration
3	Credit History
4	Purpose
5	Credit amount
6	Savings
7	Employment duration
8	Installment rate
9	Personal status
10	Debtors
11	Residence
12	Property
13	Age
14	Installment plans
15	Housing
16	Existing credits
17	Job
18	Liable people
19	Telephone
20	Foreign worker
21	Classification

Some work with classification algorithms has already been done with this dataset. The results of different classifiers used with this algorithm can be seen inworks like the ones developed by Huang, et al.[6], Erggermont, et al.[7],O'Dea, et al.[8] and Ratanamahatana and Gunopulos[9]. The approach each of the authors took was different and, as a consequence, different results can be seen in their works. In section 5.6 a comparison between the algorithms proposed in these papers and previous work with this dataset is discussed.

5CLASSIFICATION ALGORITHMS

In order to analyse the data in the dataset to build a classification algorithm some initial tests were conducted using simple classifiers from the *WekaToolkit* [11]. The dataset was presented to some of the most common classification algorithms and their results were assessed later. The results of these simple tests are shown in the Table 2.
From these results it is observed that the rate of accuracy is in the region of 70 to 78 %. Classifiers like the *J48* and O*neR*are not simple to update once they require that each time their model has to be build it is required that all model data must be provided. Here a simple update in the model will require a full evaluation which is something that is not ideal. *Naïve Bayes* and neural networks such as *Multilayer Perceptron* provide us the capabilities to update the classification model

without the need to revaluate all the data presented. Their internal structure is able to update an initial model just considering the new data available and the current model state.

TABLE 2
INITIAL CLASSIFICATION MODEL'S STUDY

Algorithm	Correct Evaluation(%)	Error (%)
J48	77.6	22.4
OneR	72.4	27.6
MultilayerPerceptron	73.5	26.5
NaïveBayes	75.6	24.4

5.1 *Multilayer Perceptron*

The *Multilayer Perceptron* is an algorithm that uses a feed forward neural network with back propagation to classify instances.
In this network a variable number of hidden layers can be used with a different number of neurons. Each neuron has a weight attributed to him and uses also a nonlinear activation function which was developed to model the frequency of action potentials of biological neurons in a brain. The most common activation functions are sigmoid and they are used in this algorithm. Another interesting property of this type of neural network is that there no connections between neuron in the same layer, however neurons are fully connected between layers and it often used more than 3 layers in the network.
The back propagation learning algorithm changes the weights in each neuron after each instance of a dataset is processed based on the amount of error in the output compared to the expected result. We represent the error in output node j in the nth data point by ej(n) = dj(n) − yj(n), where d is the target value and y is the value produced by the*MultilayerPerceptron*. We then make corrections to the weights of the nodes based on those corrections which minimize the error in the entire output, given by the equation in (1).

$$\varepsilon(n) = \frac{1}{2}\sum_{j} e_j^2(n) \tag{1}$$

Next using a method of the gradient descent it is found the change in each neuron weigth to be the value from (2)where yi is the output of the previous neuron and η is the learning rate, which is carefully selected to ensure that the weights converge to a response fast enough, without producing oscillations.

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial v_j(n)} y_i(n) \tag{2}$$

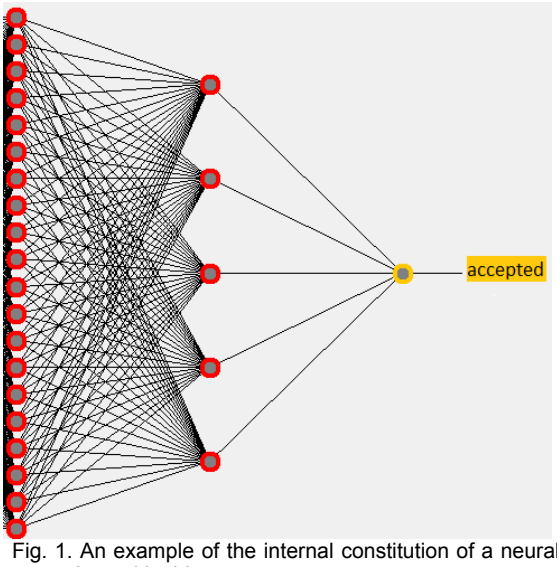


Fig. 1. An example of the internal constitution of a neural network used in this paper.

In this paper, the M*ultilayer Perceptron* used, contains an input layer with one neuron for each client attribute, a hidden layer composed of 5 neurons and an output layer with one neuron responsible for delivering the client classification. In Figure 1 it depicted the internal structure of the neural network used with the dataset from section 4.

5.2 Decision Trees

One problem that neural networks face is the fact that they consider each attribute as equal, and do not consider attributes of a given dataset that may be more relevant for the decision model than others. In this section is discussed a feature selection algorithm based in C4.5 decision trees.
Decision trees are popular methods, robust for noisy data and capable of learning disjunctive expressions. These decision trees in each of the internal nodes specify a test on some attribute from the input dataset. Each branch descending from a node corresponds to one of the possible values of the attribute specified for that node and each test results in branches that represent different outcomes of the test. The basic algorithm to induce the decision tree is a greedy algorithm that constructs decision trees in a top-down recursive manner. Additionally it implements a divide and conquer strategy to build the model.
The algorithm starts with a single node

representing all the data in the dataset. If the sample of data considered is of the same attribute class then that node becomes a leaf in the decision tree. Otherwise, the algorithm chooses an attribute that better divides the sample data into individual classes of that attributes. The process is recursive and ends when the sample data in a node is all of the same attribute class or when there is no more attributes to divide the sample data by.
The decision tree often uses an entropy-based measure as a heuristic for selecting the attribute that will best split the sample data into separate classes. In each round the algorithm computes the process above described known as the information gain for each attribute and then chooses the one with the highest information gain as the test attribute of the sample data and perform the split point. The best split point is easily evaluated considering each unique value for each attribute in the sample data as a possible split point and calculating the information gain of each one.

5.3 Feature selection algorithm

The proposed feature selection algorithms in this paper use decision trees and their proprieties to select some of the most relevant attributes in a given dataset.
The assumption that serves as a base for this algorithm is that decision trees consider the best set of attributes that classify the sample data for the upper branches in a decision tree. From this information two feature selection algorithms are proposed. Both of them use the *J48* classifier from the *Weka Toolkit*[11] with a confidence factor of 0.25 to produce a decision tree from the dataset. Then the first algorithm chooses all the attributes presented in such decision tree as important and delivers the set. Not all attributes from a dataset may be presented in a decision tree and those who are not can be considered as less important in the process of classifying instances.
The second algorithm aims to get a reduced list of the most relevant set of attributes for decision making in a dataset. Those are placed in the upper levels of the decision tree. In this case all attributes presented in the first three levels of a decision tree are selected and returned as the most important algorithms in the given dataset.

5.4 Neural networks with feature selection

From the work above some approaches are now considered to implement feature selection upon neural networks making them more aware of relevant attributes to whom special consideration should be given. The internal structure of each *MultilayerPerceptron* applied in this section has the same components detailed in section 5.1.
To accomplish feature selection upon the neural network two approaches will be considered.
The first approach uses the selection algorithm in section 5.3 that uses all attributes in a decision tree described above. Then the data is filtered and those attributes not featuring in the feature selection set are eliminated from the dataset. With the new dataset we present it to the neural network, more precisely a *MultilayerPerceptron* and train the network with the modified and normalised dataset.
The second approach uses the second feature selection algorithm presented in the section 5.3. With the given attributes from the feature selection algorithm now a special normalization of the dataset is performed. The attributes indicated from the feature selection are normalized within a range from 0 to 2 and all the other attributes are normalized within a range from 0 to 1. Neural networks are very sensible to the input data and normalizing the dataset in different ways will led the network to pay more attention to the values with greater amplitude. In the next section we will see that this last approach yields good results when compared to other alternatives and inclusive the simple *Multilayer Perceptron* algorithm.

5.5 Results from neural networks with feature selection

With the German dataset used for this project, a number of tests were made using the algorithms detailed above. In Table 3 it is presented a short summary of the results in terms of correct predictions. All tests were made using the dataset described in section 3 and a test split of 66% for training data and 33% to evaluate the behaviour of each algorithm.
With these initial results we have the normal *Multilayer Perceptron* algorithm as a base of comparison between this method and the others proposed.
Multilayer Perceptronwith Feature Selection 1 represent the first algorithm that combines the first feature selection algorithm proposed with the first evolution of the *Multilayer Perceptron* algorithm proposed in section 5.4. The test shows a decrease in the accuracy of the neural network. This can be explained with the loss of information introduced by the combination of the feature algorithm in the

dataset. From this result it is fair to conclude that reducing the dataset may not improve the client classification.

TABLE 3
RESULTS OF THE PROPOSED ALGORITHMS

Algorithm	Accuracy (%)	Error (%)
Multilayer Perceptron Simple	73.5	26.5
Multilayer Perceptron with Feature Selection 1	69.7	30.3
Multilayer Perceptron with Feature Selection 2	76.0	24.0

The second approach *Multilayer Perceptronwith feature selection 2* represents the second algorithm proposed in section 5.4 that combines the second feature selection algorithm with the *Multilayer Perceptron*. Here we see an improvement in the accuracy of the neural network. The larger range in the select attributes induces in the neural network a special attention to such attributes in relation to others leading to better results than the simple multilayer algorithm. This later algorithm also performs almost like the *NaïveBayes* in terms of accuracy.

In Table 4 we see the behavior of the *Naïve Bayes* and the last *Multilayer Perceptron* algorithm proposed in section 5.4. Here the tests conducted also used the German credit dataset but all instances of the dataset were used for training and evaluation of each algorithm.

TABLE 4
RESULTS WITH THE BEST MODIFIED MULTILAYER PERCEPTRON ALGORITHM, *J48* AND *NAÏVE BAYES*

Algorithm	Accuracy (%)	Error (%)
Naïve Bayes	75.6	24.6
J48	90.2	9.8
Multilayer Perceptron with feature selection 2	97.2	2.8

As the test shows when a case that was initially handled in the financial institution and given for learning the second algorithm proposed in section 5.4 shows a better performance than *Naïve Bayes* and the J48 algorithms. This leads to the conclusion that the proposed algorithm retains information better and when presented with the same cases it outperforms other algorithms.

In the application designed the second algorithm proposed in section 5.4 shall be used in order to classify clients and their loan applications.

5.6 Result comparison with previous work on the dataset

As stated in section 4, the dataset chosen has already been used in other papers. Unfortunately, the algorithms produced by Hung, et al.[6] and Eggermont, et al. [7] use metrics of evaluation not directly compatible with the metrics used in this paper.

In Table 5 we present a comparison betweenthe results achieved in this paper with other algorithms.

The results use 66% of the data presented in the dataset for training the classifiers and the 33% left to evaluate the answers given by each classifier. In Table 5 we compare the algorithm (a) *Combining Feature Selection and Neural Networks for Solving Classification Problems* presented by O'Dea, et al. [8],the(b)Selective Bayesian Classifier algorithm presented by Ratanamahatana and Gunopulos [9] and the last algorithm presented in this paper in section 5.4 (c) *Multilayer Perceptron with Feature Selection 2*. The first algorithm (a) uses a neural network with a reduced version of the dataset. The reduced dataset comes from attribute selection based on information theory. The second algorithm (b) also uses a reduced version of the dataset in conjunction with the Bayesian Classifier. The reduced version once again is obtained from attribute selection but unlike the previous attempt it is based on a decision tree classifier which considers the position of attributes in the tree to select those which are considered more relevant. The last algorithm, (c) *Multilayer Perceptron with Feature Selection 2* uses the full dataset, however, feature selection is used to distinguish important attributes in the neural network.

TABLE 5
RESULTS COMPARISON (A), (B) AND (C)

	Algorithm	Accuracy (%)	Error (%)
(a)	Combining Feature Selection and Neural Networks for Solving Classification Problems	76	24
(b)	Selective Bayesian Classifier	75	25
(c)	Multilayer Perceptron with Feature Selection 2	76	24

As the results show us the rate of accuracy achieved by the algorithm presented in this paper is similar to the rate of accuracy achieved by other authors in their attempt to improve other classifiers.

6SUGGESTION TO CLIENTS

6.1 Study case

After deciding upon a classification algorithm, it was felt that pure classification of client might be improved if a suggestion mechanism exists in the system. A client before applying for a loan normally uses a simulator or a conversation with a consulter from a financial institution. In this process the client may only be interested in evaluating his chances of being granted a loan from that financial institution.
In this context a suggestion model might be useful to the client and may also help the financial institution advice his client in the better set of actions he can take for improving his chances of being granted with the desired loan. Let's imagine a client that to whom a loan application was refused using the classification model present. With a suggestive algorithm he may find a solution for his problem. He would give the system an incomplete set of information of a predetermined set of attributes he cannot change and the system would calculate how changes in the not specified attributes would increase his chances to be granted with the loan. These changes might be increasing is credit amount available in the financial in a different account, reducing the amount of the loan by a percentage or even change the installment plans. With this new information the client could be advised about these new potential situations that allow him to be granted with the desired loan.

6.2 Proposed algorithm

As the proposed classification model in this system is derived from a neural network the process of building a suggestive mechanism becomes more difficult as the neural network does not provide any explanation on the results given. In fact neural networks behave much like black boxes where some input is given and some output is performed. With this approach the initial idea might be to apply a brute force algorithm on the incomplete information a client gives and test every possible outcome for some with a positive classification.
The solution given in this work however will try to be more efficient in the process of looking for positive answers. The idea is to use genetic algorithms to perform a search in the space of possible solutions and after a certain amount of time deliver the positive answers where possible to the client.
Genetic algorithms are generally constituted by:

- An initial population
- A population size
- An objective function
- Operators of selection
- Operators of variance and mutation
- Operator to substitute better individuals of a population with better ones calculated previously.

The evolution cycle of genetic algorithms is summarized in Figure 2.
In the context, in the credit data system each individual in the population will be the set of attributes that were not specified by a client. At the beginning a population of about 10 individuals is randomly generated from the possible set of solutions.

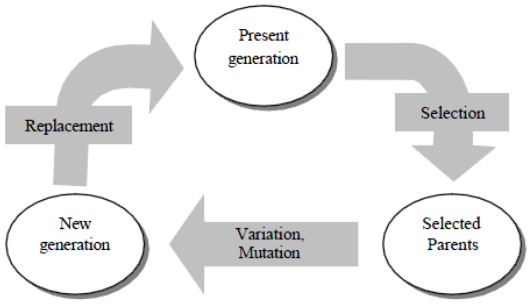


Fig. 2. Genetic algorithm evolution cycle.

According to a selection percentage defined in the beginning of the algorithm a number of individuals are selected from the population and a variance operator is performed. As each individual has the same order in the attributes a split point in that order is calculated in every pair of individual. Next, the attributes from the first individual are taken until the split point and then the order is completed with the values of the attributes of the second individual after the split point. The same process in reverse is done to second individual. This process implements the variance operator.
The mutation operator is also implemented in the selected individuals. One attribute from each of the selected individuals is selected and it isapplied a random value in the range of possible solution for that attribute.
Finally we have the objective function implemented at the expense of the classification algorithm that delivers a value above 0 in case of a good classification or below 0 in a bad classification. To obtain a

value from the neural network the set of attributes found for each individual is merged with the information given by the client and a value is generated by the classification algorithm. The next population is the 10 best individuals that were classified by the neural network with greater values from the previous and newly set of individuals.
This process is repeated a determined number of times or when all individuals in the population have positive classification by the classifier and the results are presented to the client.

7 SYSTEM CONFIGURATION

In this section we use the algorithms described earlier in this paper in order to build an agent based application to assess clients in a financial institution. This system uses an agent platform where different agents are responsible for different tasks in the system.In this system a total of 5 different types of agents were created:

- *InformationFeeder,* an agent that must gather new that and pass it into the *modelbuilder* agent that builds the decision model;
- *ModelBuilder,* an agent that, based on the second classification algorithm explained in section 4.4, builds the decision model with information received from the *InformationFeeder*. After building the model it sends it to the *DecisionAgent* and to the *SuggestiveAgent,* so they can give their answer based in the new model;
- *DecisionAgent*, an agent that must take the model built by the agent *ModelBuilder* and respond to customers enquires on complete loan applications based on its decision model;
- *SuggestiveAgent*, an agent that, with a model received from the *ModelBuilder* agent and some incomplete information from the customer, suggests him a series of favourable combinations to be successful in an loan application;
- *InquiryAgent,* an agent that makes requests to the agents *DecisionAgent* and *SuggestiveAgent* and delivers the responses to the clients.

With these 5 different types of agents the system is build and acts in a predefined way.An*InformationFeeder*is used to import a series of data into the system. Whenever new data becomes available to take into consideration this agent is created and it looks for agent of type *ModelBuilder* and delivers the data to him. The *ModelBuilder* agent, upon receiving the new data updates its decision algorithm. If it is the first time some information is given to him, then a feature selection algorithm used to determine the most relevant attributes and then those attributes are normalised in the same order as the second algorithm proposed in section 4.4. If the model already had evaluated a set of data then the new data must be normalized according to rules created the first time the model was created and the neural network should then be retrained.
When a new classification model is available the*DecisionAgent*updates its decision model with the data and normalizing rules for the input data. This agent then waits for inquiries from the*InquiryAgent* agent and then delivers the responses to him.
In a similar manner the *SuggestiveAgent*updates its classification model and also waits for inquiries from the *InquiryAgent* agent to deliver the set of suggestions based on a input to the client.
Finally the *InquiryAgent* agent is responsible for receiving the requests from an user and redirecting them to the appropriate agent the *SuggestiveAgent*or*DecisionAgent*agents. It then waits for the response and delivers it to the client. Each time a new client is introduced in the system an*InquiryAgent* agent is created and, as a consequence, more than one of these agents may exist in the system at the same time.

CONCLUSIONS AND FUTURE WORK

The algorithms described in this paper provided good results in the classification of clients for loan application in some financial institution. The proposed classification algorithm showed improvements when compared with his normal version. The suggestive algorithm also produced good results evaluating alternatives to client situations. The final system architecture proposed to integrate each individual algorithm in a distributed system poses as an example of how these algorithms can be used in real world applications
Different datasets could be used to train the classifier, also different classifiers could be improved in order to have a more comprehensive list to compare performances between each algorithm and some more work could be done allowing different *ModelBuilder*

agents that induce different decision models at the same time.

ACKNOWLEDGMENT

This work was supported in part by a Scholarship APPIA, Portuguese Association for Artificial Intelligence.

REFERENCES

[1] M. Vojtek, E. Kocenda, “Credit Scoring Methods”, Czech Journal of Economics and Finance, 2006.

[2] S. F. Eletter, S. G. Yaseen, G. A. Elrefae, "Neuro-Based Artificial Intelligence Model for Loan Decisions", *American Journal of Economics and Business Administration*, pp. 27-34, 2010.

[3] Md. S. Islam, L. Zhou, F. Li, "Application of Artificial Intelligence (Artificial Neural Network) to Assess Credit Risk: A Predictive Model for Credit Card Scoring", Msc dissertation, School of Management, Blekinge Institute of Technology, 2009.

[4] D. Simi'c, S. Simi'c, "An approach to efficient business intelligent system for financial prediction", *Springer-Verlag*, 2007.

[5] S. C. Madeira, A. L. Oliveira, C. S. Conceição, "A Data-Mining Approach To Credit Risk Evaluation and Behaviour Scoring", *Progress in artificial intelligence: 11th Portuguese Conference on Artificial Intelligence*, 2003.

[6] J. -J. Huang, G. -H. Tzeng, C. -S Ong, "Two-stage genetic programming (2SGP) for the credit scoring model", *Elsevier Inc*, 2005.

[7] J. Eggermont, J. N. Kok, W. A. Kosters, "Genetic Programming for Data Classification:Partitioning the Search Space", *Proceedings of the 2004 Symposium on applied computing* , 2004.

[8] P. O'Dea, J. Griffith, C. O'Riordan, "Combining Feature Selection and Neural Networks for Solving Classification Problems", *Intelligent exploration of the web*, Physica-Verlag GmbH, pp.389-401, 2003.

[9] C. A. Ratanamahatana, D. Gunopulos, "Scaling up the *Naïve* Bayesian Classifier: Using Decision Trees for Feature Selection", *In proceedings of Workshop on Data Cleaning and Preprocessing,*2002.

[10] Rapid-i, “RapidMiner”, http://rapid-i.com/content/view/181/190/ Acessed in 28/6/2010

[11] The university of Waikato, “Weka”, http://www.cs.waikato.ac.nz/ml/weka/ Acessed in 28/6/2010.

[12] Heaton Research, “Encog”, http://www.heatonresearch.com/encog Acessed in 28/6/2010.

[13] NeuroDimension, “NeuroSolutions”, http://www.neurosolutions.com/ Acessed in 28/6/2010.

[14] UC Irvine Machine Learning Repository, “UCI Machine Learning Repository”, http://archive.ics.uci.edu/ml/ Acessed in 28/6/2010.



Fábio Silva has a degree in computer engineering by the University of Minho in Portugal since 2009 and is currently a master student at the Department of Informatics of the University of Minho.



Cesar Analide is a Professor at the Department of Informatics of the University of Minho and a researcher of CCTC - Computer Science and Technology Center