Ant Colony Optimization for Logistic Regression and Its Application to Wine Quality Assessment

Victor-Emil Neagoe, Catalina-Elena Neghina, and Mihai Neghina

Abstract**—**This paper is dedicated to the application of swarm intelligence in the field of data mining. An Ant Colony Optimization (ACO) logistic regression model is presented with applications for wine quality assessment. The proposed ACO model may be designed to minimize either the mean absolute regression error (MAE) or the mean square regression error (MSE). The method is evaluated using the Wine Quality database (red wine) with 1599 11-dimensional samples provided by UCI Machine Learning Repository. The input features correspond to 11 physicochemical wine tests and the quality scores belong to the set {3, 4, 5, 6, 7, 8}. The best simulation variants of ACO logistic regression model have led to better performances than the classical Multiple Linear Regression (MLR) technique.

Keywords**—**swarm intelligence, Ant Colony Optimization (ACO), logistic regression, wine quality assessment.

I. Introduction

DATA mining is the analysis of large observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful for the data owner [6]. MIT Technology Review has chosen data mining as one of 10 emerging technologies that will change the world. Multiple regression modeling is a significant chapter of data mining that provides powerful and elegant methods of describing the relationships between a target variable and a set of input variables called predictors. Generally, when the target variable is continuous, we have a ***m****ultiple* ***l****inear* ***r****egression* (MLR) according to the equation

$$y= \beta_0 + \beta_1 x_1+...+ \beta_m x_m + \varepsilon \qquad (1)$$

where

– y is the target variable

– $x_1, x_{2,}..., x_m$ are the input variables called predictors

– $\beta_{0,} \beta_1, \beta_{2,}..., \beta_m$ are the regression coefficients.

– ε is the error term.

For the case when the response variable is categorical (discrete), *linear regression* is substituted by logistic regression, corresponding to the methods for describing the relationship between a categorical response variable and a set of predictor variables [2], [5], [6].

Second, *Swarm Intelligence (SI)* is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding [1]. Ant Colony Optimization (ACO) deals with artificial systems that is inspired from the behavior of real ants, which are used to solve discrete optimization problems [1], [7], [8], [10], [11]. Within this paper we shall propose and evaluate a logistic regression based on ACO principle.

Third, nowadays wine is increasingly enjoyed by a wider range of consumers. To support this growth, the industry is investing in new technologies for both wine making and selling processes. Certification and quality assessment are crucial issues within the wine industry [3]. Certification prevents the illegal adulteration of wines (to safeguard human health) and assures quality for the wine market. Quality evaluation is often part of the certification process and can be used to improve wine making (by identifying the most influential factors) and to stratify wines such as premium brands (useful for setting prices). The paper proposes an ACO based logistic regression approach for wine quality assessment where any wine is specified by a vector with physicochemical features.

The paper is structured as follows.

Second section presents the model of Ant Colony Optimization (ACO) for classification (logistic regression). It also includes an example of ACO classifier on the dataset PALM [9].

Third section is dedicated to the application of ACO/MLR regression for wine quality assessment using the Wine Quality database downloaded from the UCI Machine Learning Repository, University of California, Irvine.

II. Ant Colony Optimization Model for Logistic Regression

A. Ant Colony Optimization

In a colony of social insects, such as ants, bees, wasps and termites, each insect usually performs its own tasks independently from other members of the colony. However, the tasks performed by different insects are related to each other in such a way that the colony, as a whole, is capable of

Manuscript received September 30, 2010. The work has been funded by the Sectorial Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreements POSDRU/88/1.5/S/60203 and POSDRU/6/1.5/S/16.

Victor-Emil Neagoe is with the Polytechnic University of Bucharest, Romania, (phone: +40 721235020; fax: +40-21-4114700; e-mail: victoremil@gmail.com).

Catalina-Elena Neghina is with the Polytechnic University of Bucharest, Romania, (phone: +40 724518534; e-mail: neghina.catalina@gmail.com).

Mihai Neghina is with the Polytechnic University of Bucharest, Romania, (phone: +40 740027369; e-mail: mihai.neghina@gmail.com).

solving complex problems through cooperation. Important survival-related problems such as selecting and picking up materials, finding and storing food, which require sophisticated planning, are solved by insect colonies without any kind of supervisor or centralized controller. This collective behavior which emerges from a group of social insects has been called “swarm intelligence” [1].

In this paper we are interested in a particular behavior of real ants, namely the fact that they are capable of finding the shortest path between a food source and the nest (adapting to changes in the environment) without the use of visual information. The main idea is the indirect communication between the ants by means of chemical pheromone trails [1], [7], [8], [10], [11], which enables them to find short paths between their nest and food (Fig. 1). In nature, ants usually wander randomly, and upon finding food return to their nest while laying down pheromone trails. If other ants find such a path (pheromone trail), they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food. However, as time passes, the pheromone starts to evaporate. The more time it takes for an ant to travel down the path and back again, the more time the pheromone has to evaporate (and the path becomes less prominent). A shorter path, in comparison will be visited by more ants (can be described as a loop of positive feedback) and thus the pheromone density remains high for a longer time. *Ant Colony Optimization* (ACO) is implemented as a team of intelligent agents which simulate the ants behavior, walking around the graph representing the problem to solve using mechanisms of cooperation and adaptation.

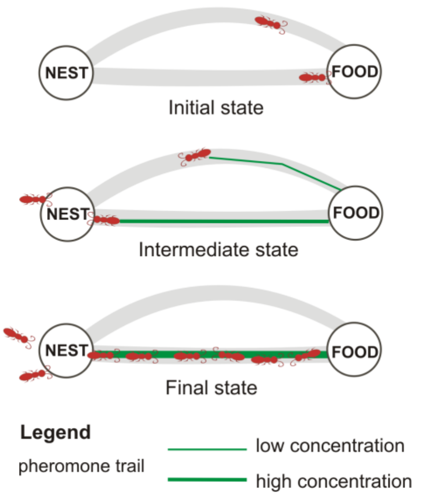

Fig. 1. Behavior of real ants as a source of inspiration for ACO model.

ACO was first introduced using the Travelling Salesman Problem (TSP). Starting from its start node, an ant iteratively moves from one node to another. When being at a node, an ant chooses to go to a unvisited node at time *t* with a probability given by

$$p^k_{i,j}(t) = \frac{[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta}}{\sum\limits_{l\in N^k_i}[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta}} \tag{2}$$

- N^k_i is the feasible neighborhood of the ant_k, containing the set of cities which ant_k has not yet visited
- $\tau_{i,j}(t)$ is the pheromone value on the edge (i, j) at the time t, (i, j)
- α is the weight of pheromone
- $\eta_{i,j}(t)$ is a priori available heuristic information on the edge (i, j) at the time t
- β is the weight of heuristic information
- $\tau_{i,j}(t)$ is given by

$$\tau_{i,j}(t) = \rho \cdot \tau_{i,j} + \sum_{k=1}^{n}\Delta\tau^k_{i,j}(t) \tag{3}$$

where

$$\Delta\tau^k_{i,j}(k) = \begin{cases} \dfrac{Q}{L_k(t)}, \mathit{if\ the\ edge}\,(i,j)\,\mathit{is\ chosen\ by\ ant}_k \\ 0, \mathit{otherwise} \end{cases} \tag{4}$$

- ρ is the pheromone trail evaporation rate ($0 < \rho < 1$)
- n is the number of ants
- Q is a constant for pheromone updating

B. ACO Classifier

The ACO algorithm for classification builds iteratively a solution to the classification task. The solution is a set of rules, each rule being associated with an output class. The ACO algorithm for classification attempts to find a set of rules of the following type

If (term1 and term2 and... and termN) then (class_k).

Each term is a composed from two triplets <*attribute*, *operator*, *value*>, which, for continuous input spaces, can be rewritten as a double inequality.

Example: $0 \le \mathrm{val_i} \le 1$, where *a* and *A* are the thresholds along dimension *i* of the current rule. These thresholds are determined by the ants during the training.

The number of terms is equal to the number of dimensions of the input vectors. As an example, for the *Wine Quality database*, the number of terms will be 11 because the vectors are 11-dimensional. For each rule there are two thresholds per dimension which will be determined.

In order to simplify the task without reducing the generality of the solution, the input space (containing all vectors) may be transformed by compression and translation so that the value of each component of the vectors falls in the interval [0...1]. In other words, each component val_i of a vector V will satisfy the relation $0 \le \mathrm{val_i} \le 1$.

The algorithm is a sequence of steps for each dimension to discover the corresponding rules, by discovering both thresholds *a* and *A* of each term.

The algorithm is generally described in the following pseudo code:

```
WHILE there are sufficiently many vectors left
  FOR iAntEpoch = 1 : number of epochs (for each epoch)
    A rule is being determined in the following steps using an ant:
    FOR iDim = 1 : number of dimensions (for each dimension)
      Computation of ETA (see B.a)
```

```
            Computation of roulette probabilities for thresholds a and A (see B.b)
            The thresholds are chosen with the help of a random variable (see B.b)
        end FOR
        Rule Pruning for the newly constructed rule (see B.c)
        Class determination for the newly constructed rule (see B.d)
        Computation of path quality (see B.e)
        Update TAU (indicating the quantity of pheromone, see B.f)
    end FOR
    Determining the best rule from all epochs
    Inclusion of that rule in the Rule List
    Elimination of the vectors covered by the new rule from the training set
end WHILE
```

At the beginning, the rule list is empty, and the training set contains all training vectors. At each iteration of the WHILE loop, a new classification rule is discovered and added to the Rule List, and the training set is adjusted by removing the vectors covered by the rule. The process is repeated iteratively until the number of vectors not covered by a rule is smaller than a certain predefined constant.

B.a. Computation of ETA (η)

ETA (the apriori information) is determined for each dimension and each vector set. Given a set of training vectors, the following steps are followed in order to compute ETA for a certain dimension:

– All vectors are projected onto that dimension.

– The quality of a cut is computed for each raster point with the help of the following formula

$$\eta_j = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{N_i}\max\left(N_{i\_left}^{(j)}, N_{i\_right}^{(j)}\right) \qquad (5)$$

- η_j = the ETA coefficient for a cut at point *j*
- K = number of classes
- $\mathrm{N_i}$ = number of vectors in *class i*
- $N_{i\_left}^{(j)}$ = number of the projections of the vectors of *class i* located to the left of the *cut j*
- $N_{i\_right}^{(j)}$ = number of the projections of vectors of *class i* located to the right of the *cut j*.

For extreme and close-to-extreme values, a check is performed to test whether there are enough vector projections towards the shorter side of the cut. For instance, a cut in zero does not split vectors from the same class, but it also does not help the ants reach a useful decision. For this type of cuts which do not bring information, the ETA coefficient is 0, thus eliminating the possibility that the ant chooses that cutting point as a threshold. However, there is still the possibility that a rule to contains *a = 0* and *A = 1* along one dimension, because of *rule pruning* described in later paragraphs.

Each ETA coefficient can have any value between 0.5 and 1. If the cut separates perfectly the classes (all the vectors of the same class are on the same side of the cut), then η= 1. If for all classes, half of the vectors are on the right side and the other half are on the second side of the cut, then η = 0.5.

ETA is the vector of coefficients of type η, representing the quality of the cut for each raster point.

Example 1

We further show an example of computing ETA, for seven vectors belonging to two classes (blue and red), already projected on the considered axis (see Fig. 2).

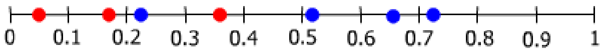

Fig. 2. Example of computation of ETA for a two class partition.

We shall further show the computation of ETA for a cut in 0.3. For each class, a measure of the cut is computed using the formula (5).

- For Red class : r = max(2,1)/3 = 2/3
- For Blue class : b = max(1,3)/4 = 3/4

The ETA coefficient is $\eta_{0.3}$ = (r + b) / 2

B.b. Roulette probabilities for thresholds a and A

The computation of roulette probabilities for thresholds *a* and *A* is performed using the formula

$$p_j(t) = \frac{[\tau_j(t)][\eta_j(t)]}{\sum_{l\in N}[\tau_j(t)][\eta_j(t)]} \qquad (6)$$

where

- N : the number of possible cuts for the next threshold
- η_j (t) : the quality of *cut j*, at time *t*
- τ_j (t) : the pheromone value on the path from the current location of the ant to the *cut j*, at the time *t*

To choose the threshold, a roulette is used during the following steps:

- Natural numbers *i* are associated with the possible thresholds
- Probabilities p_i are computed for each possible threshold, using formula *(α)*
- A random subunitary number *n* is generated
- The chosen threshold is associated with the lowest number *i* satisfying the relation $sum(p_i) > n$

After all the thresholds are computed, the rules resemble with multidimensional parallelepipeds (hyper-parallelepipeds), which may contain fewer vectors than the minimum number of vectors to be covered by a rule. Therefore, the next step of the algorithm involves *rule pruning.*

B.c. Rule pruning

Rule pruning is done by extending the *hyper-parallelepiped* until it contains enough vectors to enable the computation of the rule (path) quality. The extension is performed iteratively in small steps. Using an uniformly distributed random variable ***a***, one chooses a dimension along which the extension is performed and a direction, so that one threshold is moved towards the extremity: either *a* becomes *a – step* or *a* becomes *a + step.*

B.d. Class assignment according to a new rule

We have proposed class assignment according to a new rule taking into account the distribution of covered vectors. The class with the most vectors covered by the rule becomes the class associated with the rule. This class label (a quality score expressed by a positive integer) will be further used in computing the performance of the rule (path).

B.e. The performance measure of the rule (path)

As already mentioned, the path is the complete set of thresholds defining a rule. In the classical ACO classification algorithms, the performance measure of the path (in the design procedure) was either the percentage of correctly classified vectors, or other measure derived from that . For the quality assessment application, there are significant inequality relationships between quality classes (for example, score3 < score4< … < score8). Since then, we have chosen a path performance measure adapted to the application of quality assessment by integer scores. Thus, either the Mean Absolute regression Error (MAE), or the Mean Square regression Error (MSE) is used as a measure for design of the presented ACO model, in order to emphasize the seriousness of the error rather than the number of errors.

The best rule in all epochs, given by the ant whose path gives the minimum MAE (or MSE respectively), will be added to the Rule List.

B.f. Update TAU

The pheromone quantity used for updating the TAU matrix is directly proportional with the quality of the path, in the sense that better rules update greater quantities of pheromones. The formula is

TAU(t+1) = RHO * TAU + delta (TAU) where

- RHO is the evaporation factor
- delta (TAU) is derived from the path quality

Example 2

In Fig. 3, an ant path example is represented for a 2-d dataset with raster quanta = 0.2.

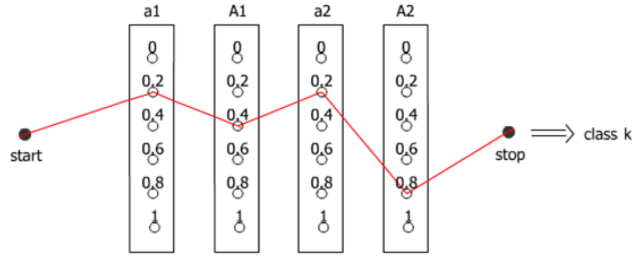

Fig. 3. An ant path example for a 2-d data set with raster quanta 0.2.

The constructed rule for the example given in Fig. 3 is

$$if\ (0.2 \leq val_1 \leq 0.4)\ and\ (0.2 \leq val_2 \leq 0.8)\ then$$

all the vectors inside the *hyper-parallelepiped* described by the above rule are associated with class k.

C. Application of ACO classification for PALM dataset

We have considered the two class dataset “Palm” [9] (see Fig. 4); the training set contains 36 vectors (18 vectors for each class); the test set has the same number of vectors as the training set.

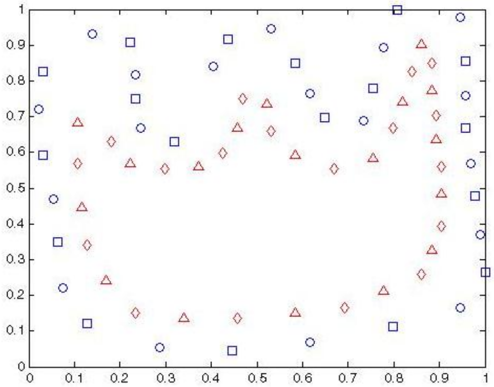

Fig. 4. A two-class dataset “Palm”; ◇ = training vector of the first class; △ = test vector of the first class, ○ = training vector of the second class; □ = test vector of the second class.

We have considered the following parameters:

- P = raster quanta = 0.02
- N = number of ants (epochs) = 15
- M = maximum number of unclassified vectors
- L = minimum number of vectors belonging to a rectangular hyper-parallelepiped
- M = L ∈ {3, 4, 5. 8, 10}

We have performed 100 simulations on the training and test sets and we have kept the best results for each subset (training/test).

The experimental results are given in Fig. 5 and Table I.

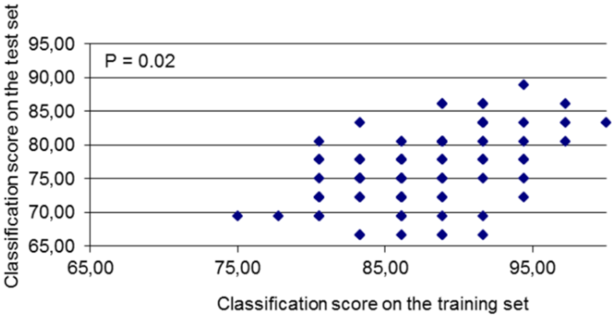

Fig. 5. Classification score on the test set as a function of the classification score on the training set for all the 100 simulations.

TABLE I. RECOGNITION SCORE (%) OF SEVERAL CLASSIFIERS FOR PALM DATABASE

Classifier	Recognition score [%]
K-Means classifier	55.56
Bayes quadratical classifier	52.78
Nearest Neighbor	66.67
Multilayer Perceptron	86.11
ACO classifier	**88.89%**

III. EXPERIMENTAL RESULTS OF WINE QUALITY ASSESSMENT USING ACO LOGISTIC REGRESSION

A. Wine Quality Database

The wine quality database is downloaded from the UCI Machine Learning Repository, Center for Learning and Intelligent Systems University of California, Irvine. We have used the dataset related to *red* ”vinho verde” wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests [3]. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).

Input variables (based on physicochemical tests) are the following:

- 1 – fixed acidity
- 2 – volatile acidity
- 3 – citric acid
- 4 – residual sugar
- 5 – chlorides
- 6 – free sulfur dioxide
- 7 – total sulfur dioxide
- 8 – density
- 9 – pH
- 10 – sulphates
- 11 – alcohol
- The output (based on sensory data):
- 12 – quality (integer score between 0 and 10).

We have divided the dataset of 1599 11-dimensional labeled vectors into the training set (2/3 of total, containing 1066 11-dimensional vectors), the validation set (1/6 of total, consisting of 266 vectors) and the test set (1/6 of total. containing the rest of 267 vectors). The 12-th dimension of each vector is the wine quality score (class label), expressed for this dataset by an integer $y \in \{3, 4, 5, 6, 7, 8\}$. The division of the database in the three sets (training and test one) has been chosen in order to preserve the quality score histogram (Fig. 6).

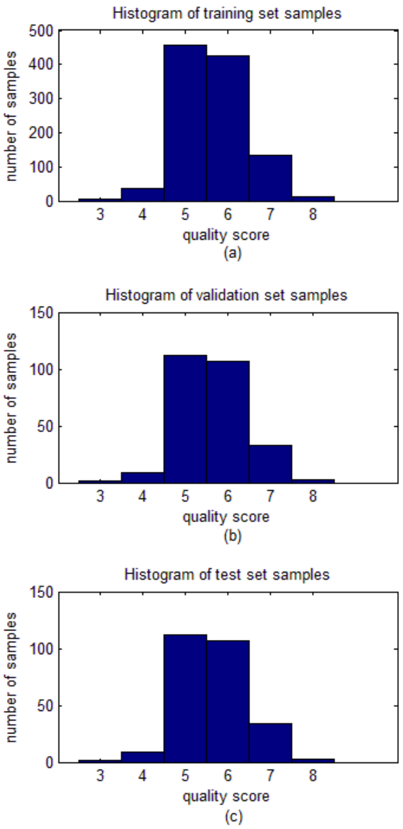

Fig. 6. Quality score histograms for the partition of red Wine Quality database. (a) training set. (b) validation set. (c) test set.

B. *ACO Logistic Regression Performances*

We have considered two variants of input normalization.

α) For each feature x_i, one performs the following transformation

$$z_i = (x_i - m_i) / \mathrm{sqrt}(\mathrm{var}\{x_i\}), \text{ where } m_i = \text{average } \{x_i\}.$$

β) Second preprocessing consists of application of the Darlington technique [4] to transform each feature distribution into a normal one, or rather “as close to a normal distribution”. After that, the resulted variable is scaled by a linear transformation in the [0,1] interval.

For the evaluation of the presented ACO logistic regression model, we have considered the *mean absolute regression error* and the *mean square regression error* as measures to evaluate the regression performance

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \quad (7)$$

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \quad (8)$$

where y_i is the correct quality score, when $\hat{y}_i$ is the estimated score by the regression model and N is the number of classes (integer quality scores). For the considered red wine dataset, $\hat{y}_i$ and y_i are positive integer numbers belonging to a set with 6 elements, namely $y_i, \hat{y}_i \in \{3, 4, 5, 6, 7, 8\}$; for the MLR, y_i and $\hat{y}_i$ are positive real numbers.

The experimental results of the wine quality assessment using Ant Colony Optimization (ACO) or Multiple Linear Regression (MLR) can be evaluated from the performances shown in Tables II, III, IV, and V.

TABLE II. ACO/ MLR PERFORMANCES FOR THE RED WINE QUALITY ASSESSMENT (VARIANT α OF INPUT NORMALIZATION)

	ACO training objective	step P	Minimum number of vectors covered by a rule L	MAE test set
MLR				0.496
ACO	MAE minimization	0.02	15	**0.468**
		0.01	11	0.491
			15	0.521
			17	0.551
	MSE minimization	0.02	15	0.483
		0.01	11	0.551
			15	0.517
			17	0.558

TABLE III. ACO /MLR PERFORMANCES FOR THE RED WINE QUALITY ASSESSMENT (VARIANT β OF INPUT NORMALIZATION)

	ACO training objective	step P	Minimum number of vectors covered by a rule L	MAE test set
MLR				0.492
ACO	MAE minimization	0.02	15	0.498
		0.01	11	0.476
			15	0.524
			17	0.476
	MSE minimization	0.02	15	0.528
		0.01	11	**0.453**
			15	0.449
			17	0.498

TABLE IV. CONFUSION MATRIX FOR ACO WITH MAE MINIMIZATION (VARIANT α OF INPUT NORMALIZATION; P=0.02; L=15; MAE=0.468)

Assigned quality score	Real quality score					
	3	4	5	6	7	8
3	**0**	0	0	0	0	0
4	0	**0**	0	0	0	0
5	100	66.67	**78.57**	37.38	23.53	0
6	0	33.33	20.54	**60.75**	58.82	100
7	0	0	0.89	1.87	**17.65**	0
8	0	0	0	0	0	**0**

TABLE V. CONFUSION MATRIX FOR ACO WITH MSE MINIMIZATION (VARIANT β OF INPUT NORMALIZATION; P=0.01; L=11; MAE=0.453)

Assigned quality score	Real quality score					
	3	4	5	6	7	8
3	**0**	0	0	0.93	0	0
4	0	**0**	0	0	0	0
5	50	88.89	**76.79**	38.32	2.94	33.33
6	50	11.11	22.32	**52.34**	52.94	33.33
7	0	0	0.89	8.41	**44.12**	33.33
8	0	0	0	0	0	**0**

IV. CONCLUDING REMARKS

1. This paper presents an Ant Colony Optimization Model (ACO) for classification and logistic regression.
2. The ACO model optimized for classification is tested on the PALM dataset (see Fig. 5 and Table 1). The best ACO simulation variants leads to the classification score of 88.89 %; this performance is better than the ones obtained by all the other tested classifiers on the same dataset (k-Means, Bayes, Nearest Neighbor, Multilayer Perceptron). The ACO simulation variant leading to the above mentioned score has obtained on the training set a recognition rate of 94.44 %.
3. An original feature of the presented ACO model for quality assessment (by comparison to other ACO classifiers) is its capability to be designed to minimize either the mean absolute regression error (MAE) or the mean square regression error (MSE). Thus, one emphasizes the seriousness of the error rather than the number of errors.
4. We have applied the ACO logistic regression model for wine quality assessment. We have chosen for experiments the Wine Quality database (red wine) with 1599 11-dimensional samples provided by UCI Machine Learning Repository; the input features correspond to 11 physicochemical wine tests.
5. The best simulation variant of ACO logistic regression has led to an mean absolute error of 0.453 when the classical Multiple Linear Regression (MLR) corresponding error is of 0.492 (Table III); this means that ACO model is better than MLR with about 8 %.

REFERENCES

[1] A. Abraham, H. Guo, and H. Liu, “Swarm Intelligence: Foundations, Perspectives and Applications”, *Studies in Computational Intelligence (SCI)*, vol. 26, Berlin: Springer, 2006, pp. 3–25.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, New York: Springer, 2006.

[3] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, “Modeling Wine Preferences by Data Mining from Physichemical Properties”, in *Decision Support Systems,* Elsevier, vol. 47, no. 4, pp. 547-553, November 2009.

[4] R. B. Darlington (1997), “Transforming a Variable to a Normal Distribution or other Specified Shape”, Available online: http://www.psych.cornell.edu/darlington/transfrm.htm.

[5] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression,* 2nd ed., New York: Wiley, 2000.

[6] D. T. Larose, *Data Mining Methods and Models,* New Jersey, Wiley, 2006.

[7] Y. Marinakis and G. Dounias “Nature Inspired Intelligence in Medicine: Ant Colony Optimization for Pap-Smear Diagnosis”, *International Journal on Artificial Intelligence Tools*, vol. 17, no. 2, pp. 279-301, April 2008.

[8] D. Martens, M. Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens, “Classification with Ant Colony Optimization”, *IEEE Trans. on Evolutionary Computation*, Vol. 11, No. 5, pp. 651-665, October 2007.

[9] V. E. Neagoe, “Decorrelation of the Color Space, Feature/Decision Fusion, and Concurrent Neural Classifiers for Color Pattern Recognition”, in *Proc. International Conference on Image Processing, Computer Vision & Pattern Recognition (ICPV'08),* Las Vegas, Nevada, USA. July 14-17, 2008, pp. 28-34.

[10] N. Z. Nejad, A. H. Bakhtiary, and M. Analoui, “A Classification Using Unstructured Rules and Ant Colony Optimization”, in Proc. of the International MultiConference of Engineers and Computer Scientists 2008, (IMECS 2008), vol. I, Hong Kong, , 19-21 March, 2008.

[11] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, “Data Mining with an Ant Colony Optimization Algorithm, *IEEE Trans. on Evolutionary Computation*, Vol. 6, No. 4, pp. 321-332, August, 2002.

Victor-Emil I. Neagoe (M’78–SM’84) was born in Pitești (Argeș, Romania) on May 31, 1947. He received the M.S. degree in electronics and telecommunications (first of series, Honor Diploma) from the Polytechnic Institute of Bucharest, Romania, in 1970, the Ph.D. degree in the same field from the same institution in 1976, and the M.S. degree in Applied Mathematics and Informatics from the University of Bucharest in 1981.

He has been a Professor of the Polytechnic University of Bucharest, Romania since 1991, where he teaches the courses of pattern recognition and artificial intelligence; digital signal processing; computational intelligence; data mining; detection and estimation for information processing and he co-ordinates several Ph.D. candidates. He has published more than 120 papers; his research interest includes pattern recognition and neural networks, biometrics, satellite image classification, sampling theory, image compression and recognition, computational intelligence.

Prof. Neagoe has been included in Who’s Who in the World and Europe 500 and he has been nominated by the American Biographical Institute for American Medal of Honor and for World Medal of Honor. Recently, he has been included in Who’s Who in the World 2011edition to be printed.