Tsakonas A., Dounias G., Shtovba S., Vivdyuk V. Soft Computing-Based Result Prediction of Football Games // Proc. of the First International Conference on Inductive Modelling ICIM'2002, Lviv, Ukraine, 20-25 May 2002, Vol. 3. P. 15-21.

Soft Computing-Based Result Prediction of Football Games

A. Tsakonas[1], G. Dounias[1], S. Shtovba[2] and V. Vivdyuk[2]

[1]University of the Aegean (Greece),
[2]Vinnitsa State Technical University (Ukraine)
www.vinnitsa.com/shtovba shtovba@yahoo.com

Abstract

Soft computing methods for result prediction of football games based on fuzzy rules, neural networks and genetic programming techniques, are proposed in the article. The models are taking into account the following features of football teams: difference of infirmity factors; difference of dynamics profile; difference of ranks; host factor; personal score of the teams. Testing shows that the proposed models achieve a satisfactory estimation of the actual game results. The current work concludes with the recommendation of soft-computing techniques as a powerful approach, either for the creation of result prediction models of diverse sport championships, or as effective data extrapolation mechanisms in case of limited available statistics.

1. Introduction

The prediction of sport game results consists an important task in bookmaking business. Besides, this task can perform as a good benchmark problem for testing diverse techniques of extrapolation and prediction under difficult conditions of limited available statistics and uncertainties of influence factors. Soft computing [1] is meant as a large variety of new powerful techniques for intelligent data analysis, which provide a suitable way for handling complexity, uncertainty and fuzziness of real-world problems. The aim of the present paper is to demonstrate an example of how to predict football game winners by applying soft-computing techniques, such as fuzzy logic, neural networks and genetic programming. Data representing the Ukrainian football championship during 10 last years are used for the creation and testing of the intelligent prognostic models applied within this paper.

2. The problem statement

From the cybernetic point of view, the task of creating football winner prediction models is reduced to that of finding out functional mapping of the form:

$$X = \{ x_1, x_2, ..., x_n \} \rightarrow D \in \{ d_1, d_2, d_3 \}, \quad (1)$$

where, X: denotes a vector of features (i.e. influence factors), such as team level, climate conditions, playing place, results of past games etc.;

D: denotes the football game result for assessment of one of the terms:

d_1: «host team's win», d_2: «draw» and d_3: «guest team's win».

3. Feature selection

From the authors' point of view, the features carrying the major influence on the game results are:

x_1**:** difference of infirmity factors (as number of traumatized and disqualified players of host team, minus the same players of guest team);

x_2**:** difference of dynamics profile (as score of host team for five last games minus score of guest team for the five last games);

x_3**:** difference of ranks (host team's rank, minus guest team's rank, in the current championship);

x_4**:** host factor (as *HP/HG - GP/GG*, where *HP* denotes the total home points of the host team in the current championship; *HG* is the number of played home games by the host team; *GP* is the total guest points of the guest team in the current championship; *HG* is the number of played guest games by the guest team);

x_5**:** personal score (as goal difference for all the games of the teams involved, within 10 years).

Note, that the above features do not consist confidential information, but it is easy for the decision maker to know the feature values before the game.

4. Fuzzy prediction model

A fuzzy logic-based model is an approximation of the «input-output» dependence, on the basis of linguistic «if-then» rules and fuzzy logic inference procedures [2]. For the creation of a fuzzy model concerning the problem represented by (1), the following steps need to be carried out [2, 3]:

* representation of the input and output variables in linguistic form;
* formalization of the expert linguistic judgements about the relationships between the inputs and the output values in a fuzzy knowledge base;
* fuzzy-model learning on the basis of a training set by tuning membership functions and rule weights.

4.1. Linguistic variables

The term-sets shown in Table 1, are used for the linguistic assessment of the input and output variables. The five-element term-set was selected for the output to raise the model practicality. It is clear, that in this case:

$$d_1 = "win" \cup "large\ win"\ ;\ d_2 = "draw"\ ;\ d_3 = "loss" \cup "large\ loss"\ .$$

The formalisation of the linguistic terms is done with the use of the Gaussian membership function model:

$$\mu^t(x) = e^{\frac{-(x-b)^2}{2 \cdot c^2}}\ , \qquad (2)$$

where, $\mu^t(x)$ represents the membership function of the variable x to term t; b and c - tuning parameters - coordinate the maximum and concentration coefficients. The initial membership function parameters, defined by experts, are shown in Table 1. The optimal values of these parameters will be calculated later, during the learning phase.

4.2. Fuzzy knowledge base

Natural language expert judgements, which tie up the features and output variable, are formalised in a fuzzy knowledge base form (Table 2). Each string in this table corresponds to one «if-then» rule, for example, the first string equals to rule:

IF x_1 is «large bench» and x_2 is «many wins» and x_3 is «leader» and x_4 is «absolute advantage» and x_5 is «slashing meeting», THEN y is «large win».

4.3. Fuzzy logical inference

The result prediction of a football game will be determined by way of solving the following system of fuzzy logic equations, which is isomorphic to the fuzzy knowledge base (Table 2).

$$\begin{aligned}
\mu_{LW}(y) = {} & \mu_{LB}(x_1) \wedge \mu_{MW}(x_2) \wedge \mu_{L}(x_3) \wedge \mu_{AA}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{TS}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{L}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{LB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{TS}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{EM}(x_5);
\end{aligned}$$

$$\begin{aligned}
\mu_{W}(y) = {} & \mu_{EB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{SB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{TS}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{LB}(x_1) \wedge \mu_{MW}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{EM}(x_5);
\end{aligned}$$

$$\begin{aligned}
\mu_{D}(y) = {} & \mu_{EB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{EM}(x_5) \vee \\
& \mu_{SB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{SM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{ShM}(x_5) \vee \\
& \mu_{LB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{TS}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{EM}(x_5);
\end{aligned}$$

$$\begin{aligned}
\mu_{L}(y) = {} & \mu_{LB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{AF}(x_4) \wedge \mu_{EM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FW}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{ShM}(x_5) \vee \\
& \mu_{SB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{M}(x_3) \wedge \mu_{A}(x_4) \wedge \mu_{ShM}(x_5) \vee \\
& \mu_{EB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{O}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{EM}(x_5);
\end{aligned}$$

$$\mu_{LL}(y) = \mu_{SB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{O}(x_3) \wedge \mu_{AF}(x_4) \wedge \mu_{EM}(x_5) \vee$$

$$\mu_{EB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_F(x_4) \wedge \mu_{ShM}(x_5) \vee$$

Table 1 - Term-sets

Vari-able	Term-sets	Membership function parameters	
		c	b
x_1	Large bench (LB)	2,55	-6
	Equal bench (EB)	2,55	0
	Short bench (SB)	2,55	6
x_2	Many losses (ML)	4,25	-15
	Few losses (FL)	4,25	-5
	Few wins (FW)	4,25	5
	Many wins (MW)	4,25	15
x_3	Leader (L)	2,76	-13
	Top - score (TS)	2,76	-6,5
	Middle (M)	2,76	0
	Bottom team (BT)	2,76	6,5
	Outsider (O)	2,76	13
x_4	Absolute failure (AF)	0,7	-2
	Failure (F)	0,7	-0,33
	Advantage (A)	0,7	1,33
	Absolute advantage (AA)	0,7	3
x_5	Shameful meeting (ShM)	8,5	-20
	Equal meeting (EM)	8,5	0
	Slashing meeting (SM)	8,5	20
y	Large loss (LL)	0,64	-3
	Loss (L)	0,44	-0,9
	Draw (D)	0,44	0
	Win (W)	0,44	0,9
	Large win (LW)	0,64	3

Table 2 - Fuzzy knowledge base

##	x1	x2	x3	x4	x5	y
1	LB	MW	L	AA	SM	LW
2	EB	FW	TS	A	SM	LW
3	EB	FL	L	A	SM	LW
4	LB	FW	TS	A	EM	LW
5	EB	FW	M	F	SM	W
6	SB	FL	TS	A	SM	W
7	EB	FW	M	F	SM	W
8	LB	MW	BT	A	EM	W
9	EB	FW	M	F	EM	D
10	SB	ML	M	F	SM	D
11	EB	FL	BT	A	ShM	D
12	LB	ML	TS	F	EM	D
13	LB	FL	M	AF	EM	L
14	EB	FW	BT	F	ShM	L
15	SB	ML	M	A	ShM	L
16	EB	FL	O	F	EM	L
17	SB	ML	O	AF	EM	LL
18	EB	ML	BT	F	ShM	LL
19	SB	FL	BT	AF	EM	LL
20	LB	ML	BT	F	ShM	LL

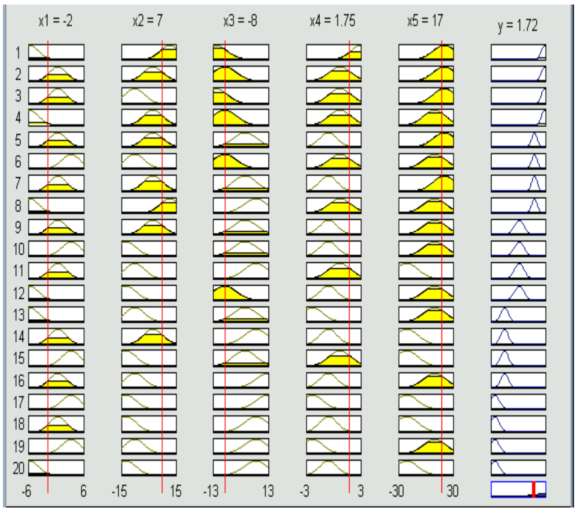

Figure 1. Fuzzy Logic Inference

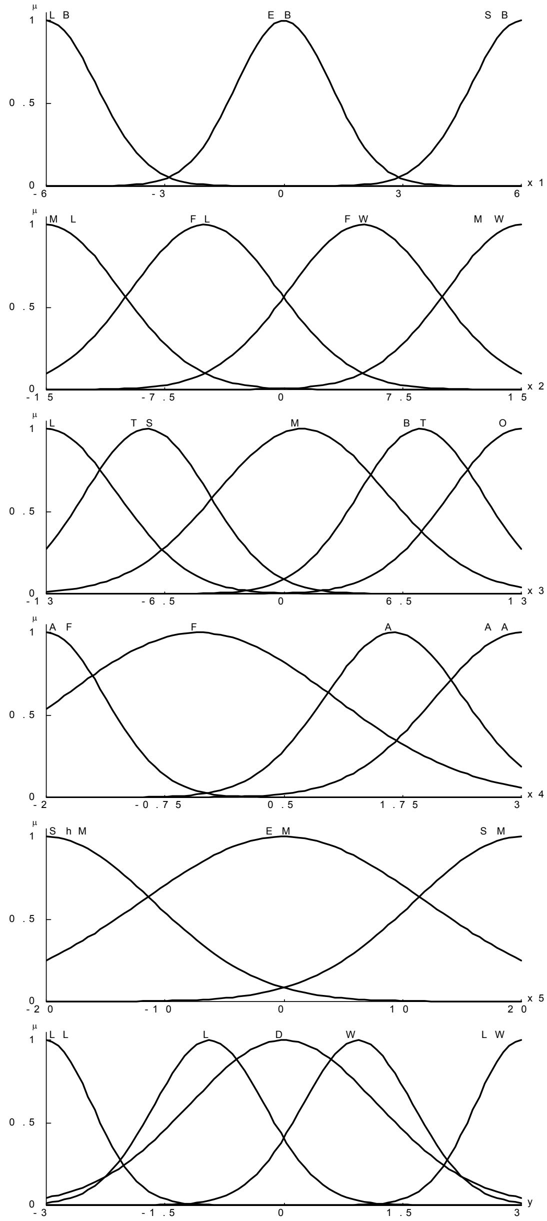
Figure 2. Optimal Membership Functions

$$\mu_{SB}(x_1) \wedge \mu_{FL}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_{AF}(x_4) \wedge \mu_{EM}(x_5) \vee$$
$$\mu_{LB}(x_1) \wedge \mu_{ML}(x_2) \wedge \mu_{BT}(x_3) \wedge \mu_{F}(x_4) \wedge \mu_{ShM}(x_5)\,. \quad (3)$$

The fuzzy logic based inference is carried out according to the following algorithm:

Step 1. Fix the feature values for the given game.

Step 2. Weaken the feature values in (2) and find out membership degrees to the linguistic terms.

Step 3. Weaken found membership degrees in fuzzy logic equations (3) and calculate decision membership degrees to terms "LL", "L", "D", "W", "LW".

Step 4. Calculate the decision in form of fuzzy set: $\tilde{y} = \bigcup_{q \in \{LL,L,D,W,LW\}} \int_{-3}^{3} min\left(\mu_q(X^*), \mu_q(y)\right)/y$.

Step 5. Calculate the crisp decision as goal difference for the game by using the center-of-gravity-based defuzzification [2]:

$$y = \frac{\int_{-3}^{3} y \cdot \mu_{\tilde{y}}(y)\, dy}{\int_{-3}^{3} \mu_{\tilde{y}}(y)\, dy}.$$

Step 6. Choose the winner by rule: $D = \begin{cases} d_1, & if\ y \in (0.5, 3] \\ d_2, & if\ y \in [-0.5, 0.5] \\ d_3, & if\ y \in [-3, -0.5) \end{cases}$.

An illustration of the proposed model and algorithm application given, with an example of outcome prediction of the game «Shahter–Metalurg vs Donetsk» (15-Sep-01, final score 3:1). The next feature values correspond to the match: x_1 = -2; x_2 = 7; x_3 = -8; x_4 = 1.75; x_5 = 17. From the results of the fuzzy logic inference, we obtain y = 1.72, which corresponds to the solution denoted as d_1. Figure 1 illustrates the fuzzy logic inference for this game result prediction in Fuzzy Logic Toolbox of MatLab's environment.

4.4. Learning the fuzzy model

Learning is the process of finding out such values of model parameters, which provide least distance between the results of modelling and the experimental data from the training set. The information of 105 football games of the Ukrainian Championship within the last 10 years was used as the training set. According to [3, 4] the tuning parameters of a fuzzy model are parameters of the membership functions and weights of the fuzzy rules. For our model the total number of these parameters is 2x24+20=68. The quantity of the tuning parameters is large, due to the use of genetic algorithms [5] for solving this non-linear optimisation problem, which consists a stochastic method of optimisation, based on the mechanisms of natural selection according to the Darwinian theory. The principal distinction of genetic algorithms from classical optimisation methods, lies in the fact that the genetic approaches do not use the notion of a derivative (gradient) during their search direction process. Genetic algorithms are based on crossover, mutation and selection operations, which allow finding out near-to-global-optimal solutions quickly, and besides no complex mathematical formulation is required, regarding the optimisation problem. The results of the learning process for the optimal membership functions and the optimal rule weighs are shown in Figure 2 and Table 3, accordingly.

Table 3 – Optimal rule weights

##	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Weight	.5	.95	.5	.63	.5	.75	.5	1	0	.22	.1	.08	.01	.29	.3	1	1	1	1	1

Table 4 – Validation of the models

Fuzzy model		Neural network		Genetic programming model	
MSE	Correct	MSE	Correct	MSE	Correct
1.60	64 %	1.71	64 %	N/A	76 %

The results of the model validation on an 175-element set (training and testing sets), represented as mean square error (MSE) of scores, as well as percentage of correct picking (win, draw, loss) are shown in Table 4. As seen in this table, the MSE is not available for the Genetic Programming model. In order to train using such an algorithm, we have the ability to define the fitness function directly as the percentage of correct classifications, thus we do not need the application of a MSE in this case.

5. Neural network based model

A neural network represents a connected set of simple computing elements so-called neurons, which try to imitate a simplified version of the behavior of the brain cell [6]. The neural output (*y*) is determined as an activation function (*f_act*) for the weighted sum of input signals (z_i):

$$y = f\_act(\sum_i z_i \cdot v_i),$$

where v_i is the weight of i-th neuron.

To construct the prediction model, we employed the multi-layer perceptron (Figure 3), using the log-sigmoid transfer function (4) for the neurons in the input layer, the hyperbolic-tangent sigmoid one (5) for the neurons of the hidden layer and the linear one (6) for the neuron(s) of the output layer:

$$y(n) = \frac{1}{1+e^{-n}}, \tag{4}$$

$$y(n) = \frac{2}{1+e^{-2n}} - 1, \tag{5}$$

$$y(n) = n, \tag{6}$$

where n is the argument of the transfer function:

$$n = w_{1,j} \cdot x_1 + w_{2,j} \cdot x_2 + w_{3,j} \cdot x_3 + \ldots + b_j . \tag{7}$$

The neuron has a bias b, which is summed with the weighted inputs to form the net input n.

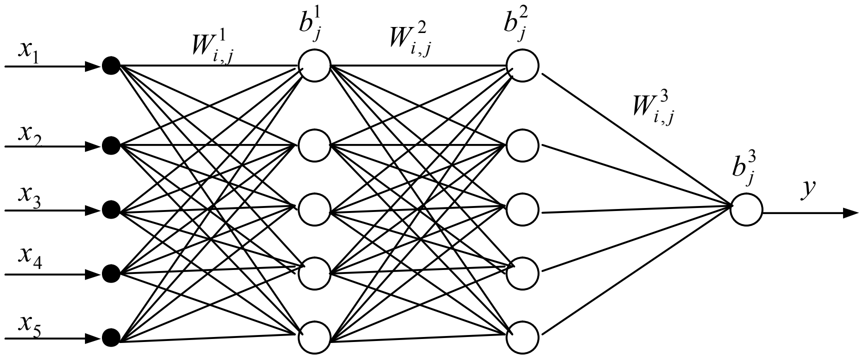

Figure 3 – Neural Network for the prediction

The learning phase of the multi-layer perceptron was carried out by the Levenberg-Marquardt backpropagation method applied on the same training set with that for the learning phase of the fuzzy model. The optimal parameters of the neural network are:

$$W^1_{i,j} = \begin{matrix} 3.1619 & 1.5569 & 0.2476 & -2.1934 & 0.8974 \\ 0.5021 & 0.1447 & 3.6600 & -0.6313 & 0.8365 \\ 0.1889 & -0.3830 & 0.8521 & -1.2201 & -0.2788 \\ 0.2234 & 0.1236 & -0.1152 & 1.1229 & -0.0261 \\ -0.8582 & -0.1034 & -1.5209 & -5.8958 & 0.0859 \end{matrix}; \quad b^1_j = \begin{matrix} -4.3134 \\ 1.8771 \\ 2.3384 \\ 1.7896 \\ -0.3430 \end{matrix}; \quad W^3_{i,j} = \begin{matrix} -0.6150 \\ 1.3220 \\ 1.6510 \\ 1.2479 \\ -0.3267 \end{matrix};$$

$$W^2_{i,j} = \begin{matrix} 2.4520 & -0.0358 & -1.1716 & 1.8782 & 0.5831 \\ -3.6167 & 2.1444 & -1.9860 & 1.5474 & 3.5628 \\ 2.4312 & 0.2541 & -0.8949 & 3.9248 & 1.3669 \\ 2.1767 & -1.2521 & -1.4786 & -0.9876 & -2.9229 \\ -1.7373 & -0.7070 & -1.5152 & -3.2672 & 0.9835 \end{matrix}; \quad b^2_j = \begin{matrix} -4.8022 \\ 0.1337 \\ -3.2475 \\ -0.9209 \\ -0.3242 \end{matrix}; \quad b^3_j = -0.1781.$$

6. Genetic programming based model

In genetic algorithms, a set of solutions, called *population*, is maintained, and the optimum solution is found by subsequent genetic operations among the members of the population that are called *chromosomes* or *individuals*. These individuals are evaluated according to the problem and a *fitness measure* is given to them. There is a number of methods for selection, such as stochastic sampling with/without replacement, tournament with/without elitist strategy etc. Crossover is the operation of exchanging genetic material between two selected individuals (named *parents)* in order to produce two new individuals (named *off-springs)*. Mutation is the random substitution of a part of the genetic material of a selected individual. When dealing with genetic algorithms, one has to determine a fixed-sized array representing the chromosome. Nevertheless, often the solution size is unknown. This drawback of genetic algorithms has led the research in extensions such as the *messy genetic algorithms* and *the genetic programming* [7-9]. In the latter concept, which is dealt in this work, individuals are represented as tree structures with variable length. The ability to represent a function into a tree opened a new way for the development of symbolic regression systems. In genetic programming, a tree node can symbolize a more or less primitive function such as addition, multiplication, Boolean algebra operators etc. Crossover in genetic programming is the exchange of sub-trees between two selected parents. Mutation may happen in two ways: the first resembles to the mutation of genetic algorithms, where a node is substituted randomly. The other, more advanced, way, called *shrink mutation* [10], selects a sub-tree of an individual, and promotes it randomly to a parental node of the same individual. The approach in this paper used the abovementioned mechanisms together with a function set that will be described below. The same data set was used as with the other experiments presented in this paper. These data were normalized here within a range [-1,1]. This normalization has shown in these GP experiments that enables faster search while keeping smaller solution sizes. As fitness measure, the number of correct score outcome was considered. This system's output is an integer. This model predicts the score outcome. That is we consider correct classification when one of the following rules are correct:

- If *forecasted_value*=>0.5 consider positive score result (home team will win)
- If *forecasted_value*<=-0.5 consider negative score result (guest team will win)
- If the absolute value of *forecasted_value*<0.5 consider zero score result (draw- no winner)

The function set used was composed by three functions (shown in Table 5). We used a population of 3000 individuals in a steady state genetic programming algorithm, using tournament selection. The steady-state approach is less greedy in computer sources, making possible the adaptation of larger populations. The tournament selection is an efficient way of selection, commonly used in steady-state genetic algorithms. As stated above, the operators used here were: reproduction, crossover, node mutation, and shrink mutation.

Table 5 - Function Set for GP Approach

Function Name	**Function Operation**	**Example**
MUL A B	Multiplication	MUL 3 4, results to 12
DIV A B	Protected Division	DIV 2 4, results to 0.5 DIV 5 0, results to 1
IFLTE A B C D	If A<B then C else D	IFLTE 2 5 6 4, results to 6

A Kill tournament is also used which replaces the worst of two randomly selected individuals. It was included a maximum age (set at 1,000 generations) for a program before substitution even if this is the best solution, in order to increase population divergence by avoiding the dominance of one good individual. The solution is presented below in pre-fix notation:

Best Solution after: 890,000 iterations

Correct Classification: Train Set: 88 out of 105, 83,8%, Test Set: 45 out 70, 64,28%, **Overall: 76%**

GP Formula: (IFLTE (DIV -78 X4) X2 (IFLTE (IFLTE X3 X5 X3 X4) X3 (DIV -89 (DIV X4 -47)) (IFLTE (DIV X1 X3) (MUL (DIV -30 -51) (DIV X4 X3)) (DIV X5 -35) X5)) (IFLTE (IFLTE X3 X4 (DIV -119 (DIV X5 -10)) 3) (MUL (DIV X1 X4) (MUL X1 -108)) (IFLTE (DIV X5 (MUL X1 -11)) X5 X1 (IFLTE (IFLTE X3 X4 -57 (DIV X4 -53)) (MUL (DIV X4 -51) (DIV (IFLTE X3 X5 -37 X4) (IFLTE -70 (MUL X1 X4) X2 X5))) (DIV X4 (MUL X1 -127)) X5)) (IFLTE (MUL (DIV X1 X4) (MUL X1 -12)) (MUL 61 (IFLTE (IFLTE (DIV -96 (DIV X4 X3)) 104 X1 6) (MUL X1 -17) X4 (IFLTE (DIV X4 -52) (DIV X5 (DIV -27 -48)) (DIV X4 (IFLTE -68 (IFLTE (IFLTE X3 X3 -53 (DIV X4 -49)) (DIV 68 -34) (DIV X4 (MUL X1 -121)) X1) X2 X5)) X5))) (DIV X4 X3) X5)))

Although scores are relatively low, it should not be neglected that the unpredictability involved in these data is high. On the other hand, the interpretability of the GP output remains relatively high, as compared with other computational intelligence solutions.

7. Prediction the championship results by the diverse models

Comparison of the diverse models for prediction of the single games is given in Table 4. Normally we expect a better classification score by the fuzzy-inference model due to the stochastic nature of many of the outcomes. However, the genetic programming approach is proved capable of achieving a high rate of classification as compared with the other models. An explanation for this may be that the genetic programming model was trained by using as fitness measure directly a pick-winner value. It is still characterized though, by comparatively heavy requirements in time and computer resources. Nevertheless, the solution proposed by a genetic programming procedure is more understandable for humans than a neural network's configuration and comparable to this of the fuzzy system. The predicting ability of models for picking winners of serial games, is based on the example of the Ninth Ukrainian Football Championship (2000-2001). Source information for the prediction is the set of results of the eight previous championships and the results of the first five tours in the current championship. The task is the prediction of the results of the remaining sixth, seventh, and twenty-sixth tours. The results of the prediction (see Table 6)., show that both the neural network and the fuzzy system were also proved capable of capturing most of the underlying trends and predicting with a high rate of success very close to the reality final rank and accumulated points.

8. Conclusions - Further Research

Further research may involve the application of statistical or entropy-based approaches, such as machine learning and support vector machines (SVM). The latter, relatively new computational intelligence approaches, could be implemented in a, common for SVM on a "±1" outcome basis, with positive values corresponding to a host-team-will-not-win outcome and negative values to a home-team-will-not-win outcome. Hybrid computational intelligent schemes might also be applied in this domain, while those systems have been proved in many cases capable of capturing nearly stochastic or chaotic processes offering a high classification and prediction rate.

Table 6 – Prediction of the championship results by diverse models

Team	Real		Fuzzy model		Neural network	
	points	rank	points	rank	points	rank
Dynamo	58	1	59	1-2	63	1-2
Shahtar	57	2	59	1-2	54	4
Dnipro	52	3	57	3	57	3
Metalurg D	51	4	53	4	63	1-2
Metalurg M	37	5	33	8	35	8
CSKA	36	6	26	10	39	6
Metalurg Z	32	7	41	7	27	10
Tavriya	30	8	49	5	37	7
Karpaty	27	9	47	6	44	5
Metalist	25	10	22	11	34	9
Kryvbas	24	11	27	9	22	11
Vorskla	20	12	3	14	3	14
Stal	11	13	21	12	10	13
Niva	9	14	11	13	13	12

References

1. Zadeh L. Applied Soft Computing –Foreword // *Applied Soft Computing*, vol.1 (2000).
2. Zimmermann H.-J. *Fuzzy Set Theory – and its Applications*.-Kluwer: Dordrecht.-1991.
3. Rotshtein A.P. Design and Tuning of Fuzzy Rule- Based Systems for Medical Diagnosis. In H.Teodorescu, A.Kandel (ed.), *Fuzzy and Neuro- Fuzzy Systems in Medicine*, CRC Press, 1998.
4. Rotshtein A, Katelnikov D., Identification of Nonlinear Objects by Fuzzy Knowledge Base // *Cybernetics and Systems Analysis,* vol.34, #5, 1998, pp 676-683
5. Gen M., Cheng R. *Genetic Algorithms and Engineering Design*.-John Wiley&Sons.-1997.
6. Looney C. *Pattern Recognition Using Neural Networks: Theory and Algorithms for Engineers and Scientists.* Oxford University Press, 1997.
7. Banzhaf W., Nordin P., Keller R., Francone F. *Genetic Programming - An Introduction*.- Morgan Kaufmann Publishers, Inc.- 1998.
8. Koza J. *Genetic Programming - On the Programming of Computers by Means of Natural Selection* .- The MIT Press.- 1992
9. Koza J., Bennett F. H., Andre D., Keane M.A. *Genetic Programming III*.- M. Kaufmann.- 1999.
10. Singleton A. Genetic Programming with C++ // *BYTE Magazine*, February, 1994.