Application of differential evolution algorithm to group bank's individual clients

Czesław Domański[1], Robert Kubacki[2]

Abstract

Grouping methods are one of the most commonly used data mining methods in banking. Their goal is to describe population of clients. They usually are a starting point for subsequent analyzes. The aim of the article is to present the results of grouping individual clients of the bank with the differential evolution algorithm. Differential evolution algorithm is an alternative to the commonly used k-means algorithm. Algorithm is generating several competing solutions in one iteration. It allows to become independent of starting vectors and to be more effective in searching for an optimal solution. Clustering was run with preselected continuous variables characterizing all individual clients (deposit, credit and investment). The calculations were run using computer program written in SAS (4GL/SQL). The differential evolution algorithm itself has been enriched with a variable that allows the selection of the optimal number of clusters. Each iteration contained proposed solutions (chromosomes) which were evaluated by the target function built on the CS measure proposed by Chou (Das et. al., 2009) . Conducted analysis showed that the algorithm correctly grouped the bank's clients.

Keywords: Clustering methods, Differential Evolution Algorithm, CS measure
JEL Classification: C38, M31, G21

1. Introduction

In today's world more and more companies have problems with effective management of available data. The gap between the amount of data that is generated, stored and the degree of their understanding is constantly growing. According to a survey conducted by IBM among the representatives of the largest banks, over 40% of them have problems with the excess of information and the lack of appropriate tools for analyzing them (Giridhar et al., 2011).

Grouping methods are effective in describing populations. Many authors have studied these methods (Everitt et al., 2011; Feoktistov and Janaqi, 2004; Gan et al., 2007).

[1] Czesław Domański, University of Lodz, Faculty of Economics and Sociology, Department of Statistical Methods, POW 3/5, 90-255 Łódź, Poland, e-mail:czedoman@uni.lodz.pl
[2] Robert Kubacki, University of Lodz, Faculty of Economics and Sociology, Department of Statistical Methods, POW 3/5, 90-255 Łódź, Poland, e-mail: robertkubacki@o2.pl

Most classic grouping algorithms have two major disadvantages:

1. Easily fall into local optima in multidimensional spaces that have multimodal objective functions
2. The efficiency of searching for a solution depends very much on the start vectors

In literature, there is a description of grouping methods as a method without a supervisor, while most traditional algorithms require a priori knowledge of the number of clusters, which means that this is not a method without the interference of an outsider. On the other hand, in many practical applications it is impossible to provide even an approximate number of groups for an unknown data set.

The limitations of classical grouping methods, including the k-means algorithm, led the researchers to search for new, more effective grouping methods. One of the directions for the development of grouping algorithms was to treat them as an optimization problem. Over the time, the paradigm of evolutionary computation, the relationship between optimization and biological evolution, has evolved. Evolutionary calculations use the power of natural selection and allow to use the computing power of computers for automatic optimization (Das et al., 2009).

2. Differential evolution algorithm – selected issues

Differential evolution algorithm is part of heuristic methods, because the goal of optimization is not to find the exact equation describing the studied phenomenon, but to search the available space for solutions. These solutions are constructed using random elements. What is more, in one iteration of the algorithm several competing solutions are created. Subsequent solutions are created using similarities to the evolutionary mechanisms occurring in nature. These are the ones that, according to the defined objective function, are the best. The characteristic feature of the differential evolution algorithm is that solutions are created on the basis of real variable vectors, not vectors coded to zero-one sequences

Since 1995 differential evolution algorithm (Storn, 1995; Storn and Price, 1997) drew practitioners' attention in optimization due to the degree of resistance, the speed of convergence and the accuracy of solutions for real optimization problems. The differential evolution algorithm has defeated many algorithms, such as genetic algorithms, evolutionary strategies and memetic algorithms (Das et al., 2016).

Suppose we have a set of objects *Np* vectors, each has *D* dimensions. In addition, we mark $\boldsymbol{P_X}$ as the current population of solutions to the optimization problem, which was created as an initial solution or at any subsequent stage of the algorithm's operation.

$$\boldsymbol{P}_{\boldsymbol{X},g} = (\boldsymbol{X}_{i,g}), \qquad i = 0,1,\dots,Np-1, \qquad g = 0,1,\dots,g_{max} \tag{1}$$

$$\boldsymbol{X}_{i,g} = (x_{j,i,g}), \qquad j = 0,1,\dots,D-1 \tag{2}$$

Index $g = 0,1,\dots,g_{max}$ denotes the generation to which the vector belongs. Each vector is assigned to the corresponding population index $i = 0,1,\dots,Np-1$. The dimensions of the vector are marked by $j = 0,1,\dots,D-1$.

The differential evolution algorithm generates mutant vectors in the next step, which will be marked as follows:

$$\boldsymbol{P}_{\boldsymbol{V},g} = (\boldsymbol{V}_{i,g}), \qquad i = 0,1,\dots,Np-1, \qquad g = 0,1,\dots,g_{max} \tag{3}$$

$$\boldsymbol{V}_{i,g} = (v_{j,i,g}), \qquad j = 0,1,\dots,D-1. \tag{4}$$

However, the vectors after crossover will be marked as follows:

$$\boldsymbol{P}_{\boldsymbol{U},g} = (\boldsymbol{U}_{i,g}), \qquad i = 0,1,\dots,Np-1, \qquad g = 0,1,\dots,g_{max} \tag{5}$$

$$\boldsymbol{U}_{i,g} = (u_{j,i,g}), \qquad j = 0,1,\dots,D-1 \tag{6}$$

The first stage, i.e. setting the initial vectors, consists in generating starting vectors. Initial parameters (for *g*=0) are set within limits that correspond to a range that is acceptable for the intended solution. Therefore, if *j*-th the search task parameter has ranges marked as $x_{min,j}$ and $x_{max,j}$ and $rand_{i,j}(0,1)$ means *j*-th realizations of a uniform distribution from the range from 0 to 1 for *i*-th vector then can be determined *j*-th component *i*-th population element, as:

$$x_{i,j}(0) = x_{min,j} + rand_{i,j}(0,1) * \left(x_{max,j} - x_{min,j}\right) \tag{7}$$

The differential evolution algorithm searches for the global optimum in *D*-dimentional continuous hyperspace. It starts with a randomly selected population *Np D*-dimensional values of parameter vectors. Each vector, also known as genome / chromosome, is a proposed solution in a multidimensional optimization issue. The next generations of solutions in the differential evolution are marked as $g = 0,1,2,\dots,g,g+1$.

The vector parameters may change with the appearance of new generations, therefore the notation for which it will be accepted, for which i-th population vector for the current generation over time (g=g) as:

$$\vec{X}_i(g) = [x_{1,1}(g), x_{i,2}(g), \dots, x_{i,D}(g)]^T \quad (8)$$

where i=1,2,…,Np.

Mutation means a sudden change in the characteristics of the chromosome gene. In the context of evolutionary computation, a mutation means a change or disorder of a random component. Most evolutionary algorithms simulate the effect of mutations through the additivity of the component generated with a given probability distribution. In the differential evolution algorithm, a uniform distribution of the vector of the form differences was used:

$$\Delta\vec{X}_{r2,r3} = (\vec{X}_{r2} - \vec{X}_{r3}) \quad (9)$$

In the differential evolution algorithm, the mutation creates a successor vector $\vec{\mathrm{V}}_{\mathrm{i}}(\mathrm{g})$ for changing the population element $\vec{\mathrm{X}}_{\mathrm{i}}(\mathrm{g})$ in every generation or iteration of the algorithm.

To create a vector $\vec{\mathrm{V}}_{\mathrm{i}}(\mathrm{t})$ for each i-th element of the current population, the other three disjoint vectors $\vec{\mathrm{X}}_{\mathrm{r}_1^{\mathrm{i}}}(\mathrm{g}), \vec{\mathrm{X}}_{\mathrm{r}_2^{\mathrm{i}}}(\mathrm{g}), \vec{\mathrm{X}}_{\mathrm{r}_3^{\mathrm{i}}}(\mathrm{g})$ are randomly selected from the current population. Indexes $\mathrm{r}_1^{\mathrm{i}}, \mathrm{r}_2^{\mathrm{i}}, \mathrm{r}_3^{\mathrm{i}}$ are mutually exclusive integers selected from a range [1,NP], which are also different from the index and the base vector. Indexes are generated randomly for each mutated vector. Then, the difference of any two of the three vectors is scaled by the number F and added to the third vector. In this way, we get a vector $\vec{\mathrm{V}}_{\mathrm{i}}(\mathrm{g})$ expressed as:

$$\vec{V}_i(g) = \vec{X}_{r_1^i}(g) + F.(\vec{X}_{r_2^i}(g) - \vec{X}_{r_3^i}(g)) \quad (10)$$

The mutation scheme shows different ways of differentiating the proposed solutions.

The crossover operation is used to increase the diversity of the population of solutions. Crossing takes place after generating a donor vector through a mutation. The algorithms of the differential evolution family use two intersection schemes - exponential and binomial (zero-one). The donor vector lists the components with the target vector $\vec{X}_i(g)$ to create a trial vector

$$\vec{U}_i(g) = [u_{1,1}(g), u_{i,2}(g), \dots, u_{i,D}(g)]^T \quad (11)$$

In exponential crossover, we first select a random integer *n* from range [0,*D*-1]. The drawn number is the starting point for the target vector from which the components are crossed with the donor vector. An integer *L* is also selected from range [1,*D*]. *L* indicates the number of components in which the donor vector is involved. After selection *n* and *L* trial vector takes the form:

$$u_{i,j}(g) = \begin{cases} v_{i,j}(g)\ dla\ j = \langle n \rangle_D, \langle n+1 \rangle_D, \dots, \langle n+L-1 \rangle_D \\ x_{i,j}(g), \qquad for\ other\ j \in [0, D-1] \end{cases} \quad (12)$$

where the intervals denote the module modulo function *D*. Integer *L* is drawn from the sequence [1,2,…,*D*] according to the following pseudocode:

L=0;

Do

{

L=L+1;

} while (rand(0,1)<CR) AND (L<D));

As a result, the probability$(L \geq v) = (CR)^{v-1}$ for any $v > 0$. Crossover rate (*CR*) is a parameter the same as *F*. For each donor vector, a new set *n* and *L* must be drawn as described above.

On the other hand, binomial crossover is carried out for each *D* variables each time, when the number selected is from 0 to 1 is less than or equal to the value *CR*. In this case, the number of parameters inherited from the donor has a very similar distribution to the binomial one. This scheme can be represented in the following way:

$$u_{i,j,g} = \begin{cases} v_{i,j,\mathrm{g}}, jeśli\ (rand_{i,j}(0,1) \leq \mathrm{C}R\ lub\ j = j_{rand}) \\ x_{i,j,\mathrm{g}}, otherwise \end{cases} \quad (13)$$

where $rand_{i,j}(0,1) \in [0,1]$ is a randomly drawn number that is generated for every j-th of the *i*-th parameter of the vector. $j_{rand} \in [1,2,\dots,D]$ is a randomly selected index that ensures that $\vec{\mathrm{U}}_{\mathrm{i,g}}$ contains at least one component from the vector $\vec{\mathrm{V}}_{\mathrm{i,g}}$.

This is determined once for each vector in a given generation. *CR* is an estimate of true probability $\mathrm{p_{Cr}}$ the event that the component of the sample vector will be inherited from the parent. It may also happen that in the two-dimensional search space, three possible test vectors can be the result of one-dimensional mating of the mutant / donor vector $\vec{V}_i(g)$ with the target vector $\vec{X}_i(g)$. Trial vectors:

a) $\vec{U}_i(g) = \vec{\mathrm{V}}_\mathrm{i}(\mathrm{g})$ both components $\vec{U}_i(g)$ inherited from the vector $\vec{\mathrm{V}}_\mathrm{i}(\mathrm{g})$

b) $\vec{U}_i{}'(g) = \vec{V}_i(g)$ one component (j=1) comes from vector $\vec{V}_i(g)$, second (j=2) from vector $X_i(t)$

c) $\vec{U}_i{}''(g) = \vec{\mathrm{V}}_\mathrm{i}(\mathrm{g})$ one component (j=1) comes from vector $X_i(g)$, second (j=2) from vector $\vec{\mathrm{V}}_\mathrm{i}(\mathrm{g})$

The last stage of the differential evolution algorithm is selection, i.e. the choice between the vector $\vec{X}_i(g)$ and a newly designated test vector $\vec{U}_i(g)$. The decision which of the two vectors will survive in the next generation g+1 depends on the value of the matching function. If the values of the matching function for the sample vector is better than the value of the target vector, the existing vector is replaced with the new vector.

$$\vec{X}_i(g+1) = \begin{cases} \vec{U}_i(g)\ dla\ \ f\left(\vec{U}_i(g)\right) \le f\left(\vec{X}_i(g)\right) \\ \vec{X}_i(g)\ dla\ f\left(\vec{U}_i(g)\right) > f\left(\vec{X}_i(g)\right) \end{cases} \tag{14}$$

where $f\left(\vec{X}\right)$ is a minimized function. The selection process consists in selecting one of two variants. The adjustment of population members improves in subsequent generations or remains unchanged, but never deteriorates.

CS (Candidate Solution) Measure proposed by Chou (Chou et al., 2004) is an objective function in this study. Group centroids are determined as the average vectors belonging to a given cluster

$$\bar{m}_i = \frac{1}{N_i} \sum_{\bar{Z}_j \in C_i} \bar{Z}_j \tag{15}$$

The distance between two points $\bar{Z}_p$ and $\bar{Z}_y$ is marked as $d\left(\bar{Z}_p, \bar{Z}_y\right)$. Then the *CS* measure can be defined as:

$$\begin{aligned} CS(k) &= \frac{\frac{1}{k}\sum_{i=1}^{k}\left[\frac{1}{|\mathrm{C_i}|}\sum_{\bar{Z}_y \in C_i} \max_{\bar{Z}_y \in C_i}\{d\left(\bar{Z}_p, \bar{Z}_y\right)\}\right]}{\frac{1}{k}\sum_{i=1}^{k}\left[\min_{j \in k, j \neq i} d\left(\bar{m}_i, \bar{m}_j\right)\right]} \\ &= \frac{\sum_{i=1}^{k}\left[\frac{1}{|\mathrm{C_i}|}\sum_{\bar{Z}_y \in C_i} \max_{\bar{Z}_y \in C_i}\{d\left(\bar{Z}_p, \bar{Z}_y\right)\}\right]}{\sum_{i=1}^{k}\left[\min_{j \in k, j \neq i} d\left(\bar{m}_i, \bar{m}_j\right)\right]} \end{aligned} \tag{16}$$

The measure is a function of the ratio of the amount of intra-group dispersion and the separation between groups. The *CS* measure is more effective at clusters with different density and / or different sizes than other measures.

3. Design of the study

The database of commercial bank clients was used for the study. It has been limited to the part of the population for which the actions taken will translate in the maximum way into business benefits. In particular, clients meet the following criteria: individual clients with active products, aged from 18 to 75 years, not being bank employees, with positive marketing consent, without delays in repayment of loan products.

As for the variables used for the study, the choice was not accidental. Variables selected for this study can be evaluated for each customer regardless of whether they have deposit, credit or investment products. Pre-processing of data allowed to eliminate outliers from the studied population. Due to the strong right-side skewness of the variables, a transformation was made by adding a constant 0.001, and then their logarithmisation. As a result, the resulting distributions of variables are more symmetrical.

The final set of variables that took part in the study is presented below:

- ZM1 (DEPOZYTY) - Total funds on accounts and deposits in thousands of PLN
- ZM2 (INWESTYCJE) - Total funds in investment products in thousands of PLN
- ZM3 (LUDNOSC) - number of inhabitants, based on the city from the correspondence address and data published by the Statistics Poland
- ZM4 (KREDYTY) - amount of bank loans taken in thousands of PLN
- ZM5 (SALDO_BIK) - balance for repayment on credit products outside the bank, based on inquiries from BIK in thousands of PLN
- ZM6 (AVG_TRN_INCOMING_ALL_3M) - average monthly income on customer's accounts in the last 3 months in thousands of PLN
- ZM7 (AVG_TRN_INCOMING_CLEAN_3M) – cleaned average monthly income on customer's accounts in the last 3 months in thousands of PLN.
- ZM8 (AVG_TRN_OUTGOING_ALL_3M) - average monthly outflows from customer accounts in the last 3 months in thousands of PLN
- ZM9 (AVG_TRN_OUTGOING_CLEAN_3M) – cleaned monthly average outflows from customer accounts in the last 3 months in thousands of PLN.

- ZM10 (AVG_TRN_OUT_DEBIT_3M) - average monthly transaction amount on the debit card from the last 3 months in thousands of PLN
- ZM11 (AVG_TRN_OUT_CREDIT_3M) - monthly average amount of credit card transactions from the last 3 months in thousands of PLN
- ZM12 (WIEK_LATA) - customer's age in years
- ZM13 (STAZ_LATA) - customer experience in years

Table 1 outlines constants used in the algorithm.

Table 1. Constants used in the study.

Constant	Value	Description of the constant
LZ	13	Number of variables describing the client
LC	13	Number of chromosomes
LK	15	Maximum number of clusters
SA	0.2	Constant activation of the vector
F	0.7	Mutation operator
Iterations	15	Number of iterations
CR	1	Crossover rate

Source: the author's own calculations.

For the purpose of optimizing number of centroids dimensional matrix is created $MR_{c,k,z}$, where c means the number of chromosomes, k means the number of clusters, z means the number of variables. Number of variables is increased by 1. An additional variable is used to store information on whether the cluster is active or inactive in the given iteration (Das et al., 2008). Values for individual matrix elements are generated according to the formula (7). An additional variable indicating focus activation is determined based on the rule: If the randomly generated number from the range 0 to 1 is smaller than the activation constant (SA) then the variable takes the value 0, otherwise it takes the value 1.

4. Results of empirical analyses

The smallest value of the CS function in the fifteenth iteration was obtained for chromosome number 3. This solution was chosen as the optimal solution.

Table 2 contains the characteristics of chromosome 3, which divided the surveyed population of the bank's clients into 9 groups (the maximum number of groups on which the population could be divided into 15).

Table 2. Numbers and share of groups for chromosome 3.

Group	No of Clients	% of total
8	92 109	45.71%
4	44 545	22.11%
6	29 047	14.41%
3	20 003	9.93%
5	5 476	2.72%
14	3 582	1.78%
1	2 839	1.41%
15	2 075	1.03%
12	1 832	0.91%
SUM	201 508	100.00%

Source: the author's own calculations.

The results of grouping in Table 2 indicate that the distinguished groups are characterized by nonequal distribution of the number of clients in groups. Group 8 is more selective and gathers 45.71% of clients, group 4 contains 22.11% of clients, and the third group 6 includes 14.41% of clients. The three mentioned groups gather over 80% of the surveyed population.

More detailed characteristics of the distinguished groups of clients are presented in the table 3, which contains average values of features in individual groups. The data presented in table 3 indicate that individual groups differ from each other. Thanks to the knowledge of average values for particular groups, it is possible to indicate groups of transactionally active customers (groups 14,5,6) and customers who use accounts less frequently (group 3,8,4,1). The most-affluent group of customers with very high means is without a doubt group number 14.

Table 3. Average values of variables ZM1-ZM13 for clusters obtained by the differential evolution algorithm

Cluster	ZM1	ZM2	ZM3	ZM4	ZM5	ZM6	ZM7	ZM8	ZM9	ZM10	ZM11	ZM12	ZM13
8	8	0	452	137	88	5	4	5	3	0	0	43	5
4	2	5	273	5	1	5	4	5	4	0	0	43	6
6	20	10	500	182	146	22	17	22	17	1	0	42	6
3	28	4	453	7	45	1	1	1	0	0	0	47	6

5	60	61	627	325	0	26	20	27	19	0	1	41	6
14	113	97	758	414	144	113	84	106	73	2	1	42	6
1	20	67	473	223	114	4	3	4	2	0	0	43	6
15	24	48	321	264	10	10	8	7	5	0	0	41	5
12	0	2	131	7	117	15	11	18	15	0	0	41	3

Source: the author's own calculations.

Thanks to the use of the differential evolution algorithm to group the bank's clients, we can get information on how many natural groups exists in a short time. Moreover, the number of groups has been calculated, not imposed in advance. The algorithm evaluated and compared obtained results for other candidate solutions in subsequent iterations, recognizing according to the values of the objective function that the optimal division of this group of customers contains 9 clusters.

5. Conclusions

The differential evolution algorithm is a promising approach to optimization, because it generates a whole set of solutions that can be easily adapted to carry out the optimization again. The fact of keeping a set of solutions, not only the best solution, allows faster adaptation to new conditions using the previously made calculations. It is resistant in terms of the choice of parameters as well as the regularity in which it finds the global optimum. Algorithm is a direct search solution method, versatile enough to solve problems whose objective function lacks the analytical description needed to determine the gradient. The algorithm is also very simple to use and modify.

Evolutionary algorithms, in particular the differential evolution algorithm do well with continuous variables when grouping clients. Customers from particular groups can be synthetically described by the mean vector for variables used in clustering. They allow to effectively separate customers with the same basket of products, but differing in the level of individual variables.

Bibliography

Chou, C. H., Su, M. C., & Lai, E. (2004). A new cluster validity measure and its application to image compression. Pattern Analysis and Applications, 7(2), 205-220.

Das, S., Abraham, A., & Konar, A. (2008). Automatic clustering using an improved differential evolution algorithm. *IEEE Transactions on systems, man, and cybernetics*-Part A: Systems and Humans, 38(1), 218-237.

Das, S., Abraham, A., & Konar, A. (2009). *Metaheuristic clustering* (Vol. 178). Springer.

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution–an updated survey. *Swarm and Evolutionary Computation*, 27, 1-30.

Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis: Wiley series in probability and statistics*.

Feoktistov, V., & Janaqi, S. (2004). New strategies in differential evolution. In *Adaptive Computing in Design and Manufacture* VI (pp. 335-346). Springer, London.

Gan, G., Ma, C., & Wu, J. (2007). *Data clustering: theory, algorithms, and applications* (Vol. 20). Siam.

Giridhar, S., Notestein, D., Ramamurthy, S., & Wagle, L. (2011). Od złożoności do orientacji na klienta. *Executive report, IBM Global Business Services*.

Storn, R. (1995). Differrential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. *Technical Report, International Computer Science Institute*, 11.

Storn, R., & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4), 341-359.