Differential Evolution For a Financial Clustering Problem

Dalvir Singh Mandara: 1423101

University of Birmingham

1. Introduction

Data classification, or clustering, has always been at the heart of intelligent data analytic methods due to it's vast applications. Basel II dictates that banks are required to have capital in place to cover losses from both expected and unexpected economic events. Classification can be used to to assign credit risk to a client so that they fall into a particular group depending on the probability that they will default on their payments in the following year. Depending on the constraints used, this problem can become NP-hard, however, heuristic optimisation methods have successfully been used to overcome this particular problem. One such method that falls under the hat of evolutionary computation is differential evolution (DE). DE has been shown to be more effective and reliable than other heuristic methods such as genetic algorithms and particle swarm optimisation when applied to the problem of assigning credit risk.

2. Problem Description

The capital that banks are required to have in place to cover unexpected losses is called regulatory capital. In order for banks to calculate this regulatory capital they need to classify their clients into different classes based on their credit risk which is determined by the probability that the client will default on payments in the following year. Once these groups have been determined, the mean probability of each group is calculated and assigned to each member of that group which acts as a label to differentiate the groups from one-another. In this setup each client in a group has the same mean probability of default (pooled-PD) from which the amount of regulatory capital required and the lending rates given to that particular group is calculated. DE can be used to determine the optimal risk groups based on the probability of defaulting. It is extremely important that banks optimally classify clients and avoid missclassification in order to comply with the many constraints that Basel II prescribes and to ensure that the amount of regulatory capital required is accurately calculated.

3. Differential Evolution

DE is a stochastic population based evolutionary algorithm introduced by Storn and Price in 1996 [2]. It has a similar setup to that of genetic algorithms (GA) except GA does cross-over and then mutation whereas DE does mutation followed by cross-over. It is assumed that every decision variable in the optimisation problem has a lower and upper bound. All evolutionary algorithms have a population, in differential evolution the population size is specified as NP and there are D decision variables

$$\vec{X} = x_1, x_2, \dots x_{D-1}, x_D.$$

It is not trivial to decide on the population size NP, initially it was suggested that $5D$ or $10D$ could be suitable population sizes however more recently it has been shown that smaller population sizes can achieve similar results as $NP = 50D$ and $NP = 1000D$.

Initialisation is usually done by randomisation. When you look at the population you have a 2 dimensional matrix with D number of columns and NP rows, every element in the matrix is initialised randomly as follows

$$x_{j,i,0} = x_{j,min} + rand_{i,j}[0,1] \cdot (x_{j,max} - x_{j,min}).$$

where each random number $rand_{i,j}[0,1]$ is recomputed for each element of the matrix. Once the initial population has been generated, fitness evaluation is done on the population.

The next operation to do is mutation. For each vector (i.e for each member of the population) you select three other solution vectors randomly (parameter vectors) and evolve the the i'th member in the following way (self-referential mutation)

$$\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F \cdot (\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}).$$

The index i is the index for the population member so the index i runs from 1 to NP as the population size is NP. Additionally the vectors

$$r^i_{1,G}, r^i_{2,G}, r^i_{3,G}$$

are three different solution members coming from the population and G is the generation number. The total population size is NP, from the population size you exclude the i'th member after-which you randomly select three other members from the remaining population. So the three parameter vectors are different from the i'th member. The first vector $r^i_{1,G}$ is known as the base vector, and then you make use of the other two randomly selected vectors to form the difference

$$r^i_{2,G} - r^i_{3,G}.$$

This is perhaps where the name differential evolution comes from. The scale factor F was initially a constant, but nowadays it can be adapted using adaptive DE methods.

An advantage of DE over other evolutionary algorithms is that you do not need to specify the mutation step size parameter (the sigma value), for example, algorithms that make use of Gaussian mutation require the specification of the sigma value. In DE the difference vector does the job of the sigma value for us and the scale factor F is a lot easier to tune than the sigma value in Gaussian mutation. If there is a search space and the values are randomly initialised over the search space, initially the population members may be well separated which means the difference vector will have a reasonably large magnitude. When the evolutionary algorithm evolves over generations the population members tend to move to a good solution region, this causes the members to become closer to each other which naturally decreases the the magnitude of the difference vector. In this fashion DE is naturally altering the search behaviour from exploration to exploitation as the population evolves. This is where the name self-referential mutation comes from, the magnitude of the difference vector is determined from the population itself.

After mutation the cross-over operation is next. The binomial cross-over method is most commonly used to produce the trial offspring and each element of the offspring vector $\vec{U}_{i,G}$ is given as follows

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if } rand_{j,i}[0,1] \leq CR \text{ or } j = R. \\ x_{j,i,G}, & \text{otherwise.} \end{cases} \quad (1)$$

where R is a random integer from $1 \ldots D$ and CR is the cross-over probability which is traditionally pre-specified but in more recent versions it has been adaptive.

The final stage is selection. The vector $\vec{X}_{i,G+1}$ is obtained by comparing the parent vector $\vec{X}_{i,G}$ to the trial offspring $\vec{U}_{i,G}$, if the objective function value of the trial offspring is smaller than or equal to that of the parent then the offspring is selected, otherwise the parent vector is selected (assuming this is a minimisation problem). In this setup there is a survival of the fittest kind of competition between the parent and the offspring, only one of them is admitted into the next generation to prevent the difference vector from collapsing to 0 too early on. Finally, note that there are various mutation and cross-over equations that can be used.

The Rastrigin function is a non-convex multi-modal function that is used to benchmark optimisation algorithms. It has many local optima and a global minimum at (0,0) in the 2 dimensional plane. An algorithm that can find the local optimum of this function can be considered effective since due to the construction of the function this is not a trivial task. The Rastrigin function is given by

$$f(x) = An + \sum_{i=1}^{n} [x_i^2 - Acos(2\pi x_i)].$$

Figure 1 shows what the function looks like and figure 2 shows the results of an implementation in Matlab using DE to find the global minimum of this function over 200 iterations. It can be seen that after 30 iterations the global minimum of 0 was found fairly quickly.

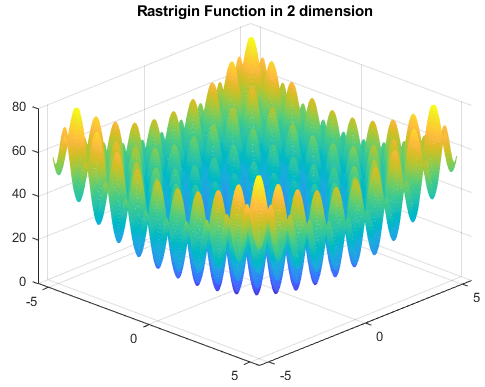

Figure 1

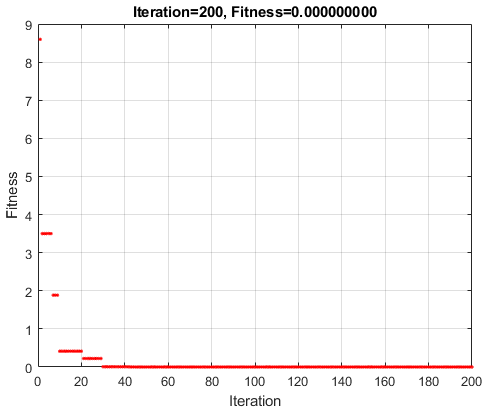

Figure 2

4. Credit Risk Assignment with DE

The assignment of credit risk to clients based on their PD is sometimes called the PD-bucketing problem. This case study focuses primarily on Italian banks which as of now currently use the k-means algorithm to tackle the PD-bucketing problem. The k-means algorithm has various drawbacks one being its susceptibility to getting stuck in local optima, as we saw from the optimisation of the Rastrigin function DE does well not to get stuck in local optima. The question for the banks is how to determine the optimal PD buckets. There are variety of constraints involved in this optimisation problem, the following three form the objective functions [1].

$$min \sum_{b} \sum_{i \in bucket_b} (\overline{PD_b} - PD_i)^2$$

$$min \sum_{b} \sum_{i \in bucket_b} |EL(\overline{PD_b} - EL(PD_i)|$$

$$min \sum_{b} \sum_{i \in bucket_b} |C(\overline{PD_b}) - C(PD_i)|$$

Where $\overline{PD_b}$ is the pooled-PD (mean PD for a bucket), $EL(PD)$ is the expected loss, and $C(PD)$ is regulatory capital. These constraints are set up with the view that banks should chose to minimise the within group variance, minimise the total error made in the measurement of the expected loss (EL) and minimise the total error made in calculating the regulatory capital (C). Amongst these constraints there are also many others prescribed by Basel II such as the requirement that banks have at least 7 buckets and no single bucket can contain more than 35% of the banks total customers.

4.1. Characterisation

As discussed earlier, this problem is a classification problem which cannot be solved with regular linear/quadratic programming with Lagrange multipliers due to way the objective functions and the constraints are to be handled[1].

4.2. Representation

For this problem consider $(g-1)$ thresholds (candidate soutions) that are required to separate all individual PD's (each over the range (0,1) since they are probabilities) into g buckets [1] that minimise the three error functions discussed above. Define a real valued vector of size $(g-1)$, and the interval between 2 consecutive vector components $[b_i, b_{i+1}]$, all clients with a PD value within this range are then assigned to bucket $i+1$ [1]. The first and last bucket intervals are defined as $[\min(PD_i), b_1]$ and $[b_{g-1}, \max(PD_i)]$ respectively.

4.3. Initialisation

When describing DE earlier it was said that the population is usually initialised randomly, however, for this particular problem plain randomisation could lead to infeasible solutions. To overcome this the population was initialised by randomly generating candidate solutions until all solutions in the population satisfy the constraint that each bucket must contain at least one solution.

4.4. Constraint Handling

A penalty based technique is used to handle constraints where a penalty term is added to the fitness value for those candidates that defy the constraints[1].

5. Discussion

Krink et al. [1] implemented DE for this particular problem and also compared it to the performance of other heuristic methods such as genetic algorithms, particle swarm optimisation and k-means for bucket sizes g = 7, 10, and 15. The bucket sizes where chosen to comply with the Basel II condition that there must be at least 7 risk categories.

Their findings show that when minimising the within-group variance (the first objective function given above), DE had a much lower standard error and a smaller 90th percentile than that of GA, PSO and k-means. This indicates that DE is more accurate and reliable than the other algorithms. Additionally, these results are also consistent when considering the second two objective functions, minimising the error in the estimation of the expected loss and regulatory capital. Another clear advantage of DE over the other algorithms is that it requires little or no parameter tuning. Additionally, DE always converged to a solution remarkably faster than the other heuristic algorithms considered in the study. As an example Krink et al. [1] found DE to be up to 50 times faster than the k-means algorithm when minimising the within-group variance.

Finally, figure 3 from Krink's paper [1] shown below shows the speed of convergence of DE, GA and PSO for the minimisation of mean expected loss over 30 runs [1] (the same pattern is also observed for the mean regulatory capital). It is clear that DE is a superior search heuristic compared to GA and PSO and could successfully be used by the banking industry to optimally classify clients into risk categories.

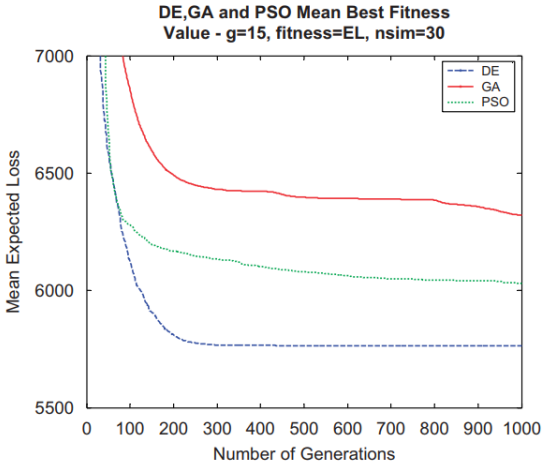

Figure 3

References

[1] T. Krink, S. Paterlini, and A. Resti. Using differential evolution to improve the accuracy of bank rating systems. *Computational Statistics & Data Analysis*, 52(1):68–87, 2007.

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