A decorative vertical grid pattern on the left side of the slide, consisting of a series of small squares in varying shades of teal and green.

# Preserving Patterns for Categorical Encoding using Soft Computing

Angel Kuri

Instituto Tecnológico Autónomo de México  
[akuri@itam.mx](mailto:akuri@itam.mx)

# Basic Background

Our aim:

To be able to apply metric clustering algorithms to mixed (numerical and categorical) data bases

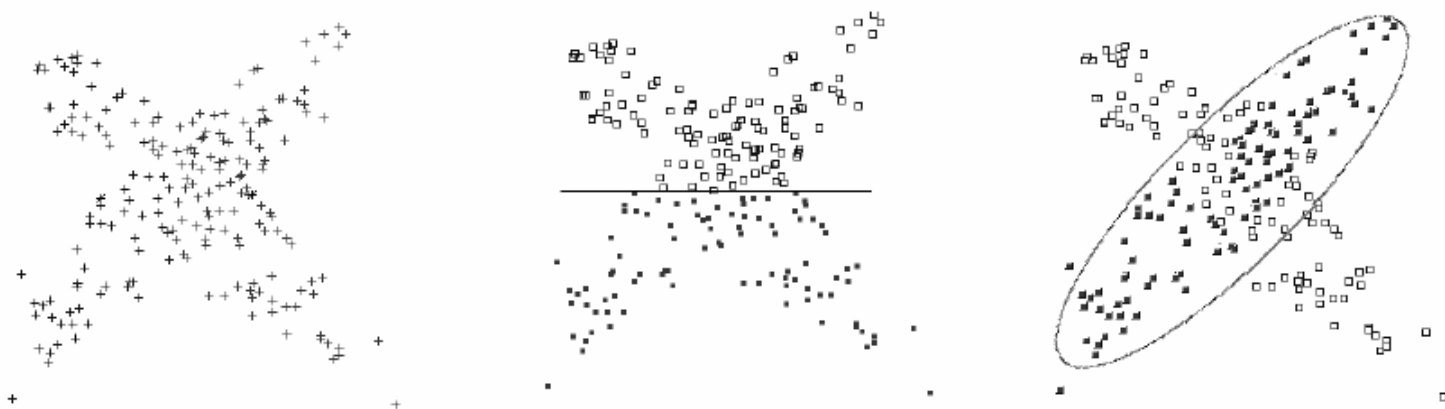
The problem:

How to assign numerical codes to the instances of categorical variables **without** affecting the patterns present in the data.

# Clustering with Metric Algorithms

Clustering is probably one of the most important issues in data mining.

The number of clusters is usually defined by the user, giving, right away, rise to interesting problems.



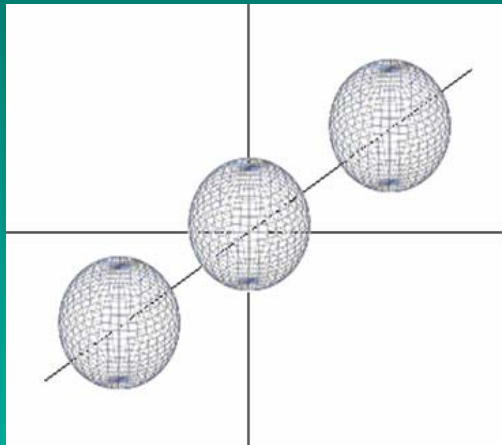
(a) Dataset  $X$  to be clustered

(b) Possible clustering of  $X$  for  $k = 2$

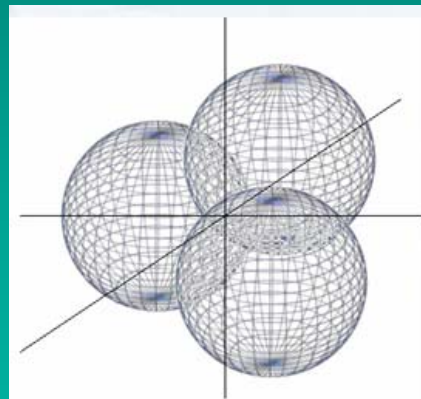
(c) Another possible clustering of  $X$  for  $k = 2$

# Metric Clustering

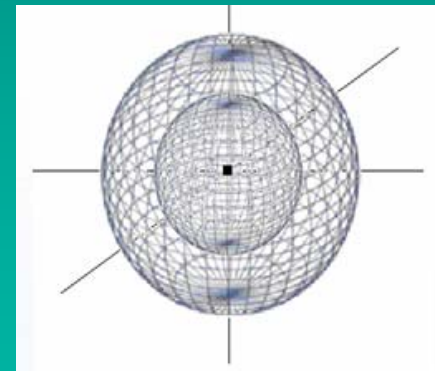
Depending on the form of the clusters its determination may become a hard problem



Not so hard



Harder



A real puzzle

# Clustering Algorithms

Clustering has been attempted using a wide variety of algorithms.

For example:

- K-Means

- Fuzzy C-Means

- Self-Organizing Maps

- Entropy Based Clustering

# An Example of a Mixed DB

| Age | Place  | Studies | Race       | Sex | Salary     |
|-----|--------|---------|------------|-----|------------|
| 35  | South  | 24      | Indian     | F   | 100,702.22 |
| 37  | South  | 4       | White      | F   | 77,770.95  |
| 24  | South  | 10      | White      | M   | 2,120.13   |
| 44  | South  | 20      | White      | M   | 9,187.65   |
| 59  | North  | 24      | Indian     | F   | 12,834.32  |
| 57  | North  | 5       | Amerindian | M   | 6,927.62   |
| 43  | North  | 8       | Amerindian | F   | 5,174.10   |
| 28  | Center | 8       | White      | F   | 2,877.14   |
| 62  | North  | 15      | White      | M   | 12,118.82  |
| 68  | Center | 23      | Amerindian | M   | 11,180.07  |
| 50  | North  | 8       | Other      | M   | 31,366.48  |
| 28  | Center | 20      | Indian     | F   | 2,996.56   |

However, if some of the attributes of the DB are NOT numerical the previous algorithms are no longer applicable. One possibility is, of course, to apply *conceptual* methods where metric spaces are not assumed.

But the advantages and objectivity inherent in numerical algorithms has led to attempt their generalization by encoding the categorical attributes, somehow.

# Previous Approaches

1. Replace every instance of a categorical attribute by an arbitrary number. For example, “South”  $\rightarrow$  1, “North”  $\rightarrow$  2 and “Center”  $\rightarrow$  3; or “Man”  $\rightarrow$  0, “Woman”  $\rightarrow$  1.

This is simple enough but it is a very bad policy because the mathematical properties of the clusters manifest as patterns. These are arbitrarily inserted/deleted from the DB using the naïve encoding just mentioned.

# Previous Approaches

2. Replace the categorical variables by a set of pseudo-binary values.

| AGE | PLACE_01 | PLACE_02 | PLACE_03 | STUDIES | RACE_01 | RACE_02 | RACE_03 | RACE_04 | SEX_01 | SEX_02 | SALARY   |
|-----|----------|----------|----------|---------|---------|---------|---------|---------|--------|--------|----------|
| 22  | 0        | 0        | 1        | 15      | 0       | 0       | 1       | 0       | 0      | 1      | 906.32   |
| 25  | 0        | 1        | 0        | 19      | 0       | 0       | 0       | 1       | 1      | 0      | 32431.18 |
| 69  | 1        | 0        | 0        | 22      | 0       | 0       | 0       | 1       | 1      | 0      | 14551.28 |
| 38  | 0        | 1        | 0        | 4       | 0       | 1       | 0       | 0       | 1      | 0      | 17394.48 |
| 68  | 0        | 0        | 1        | 12      | 0       | 0       | 1       | 0       | 0      | 1      | 765.60   |
| 45  | 0        | 0        | 1        | 15      | 0       | 1       | 0       | 0       | 0      | 1      | 1194.14  |
| 52  | 0        | 1        | 0        | 19      | 0       | 0       | 0       | 1       | 1      | 0      | 11807.69 |
| 26  | 1        | 0        | 0        | 20      | 0       | 1       | 0       | 0       | 1      | 0      | 2715.39  |
| 27  | 0        | 0        | 1        | 4       | 1       | 0       | 0       | 0       | 1      | 0      | 209.30   |
| 30  | 0        | 0        | 1        | 7       | 1       | 0       | 0       | 0       | 1      | 0      | 897.83   |
| 31  | 0        | 0        | 1        | 19      | 1       | 0       | 0       | 0       | 1      | 0      | 40738.45 |
| 43  | 0        | 0        | 1        | 19      | 1       | 0       | 1       | 0       | 1      | 0      | 18390.79 |
| 43  | 0        | 1        | 0        | 10      | 0       | 0       | 1       | 0       | 0      | 1      | 72304.83 |



# Previous Approaches

| PLACE_01 | PLACE_02 | PLACE_03 |
|----------|----------|----------|
| 0        | 0        | 1        |
| 0        | 1        | 0        |
| 1        | 0        | 0        |
| 0        | 1        | 0        |
| 0        | 0        | 1        |
| 0        | 0        | 1        |
| 0        | 1        | 0        |
| 1        | 0        | 0        |
| 0        | 0        | 1        |

In the table the attribute “Place” has been replaced by the variables “Place\_01”, “Place\_02” and “Place\_03”. The values of the instances “South”, “North” and “Center” have been replaced by “1” or “0” as appropriate. The type of coding system selected implies a subjective choice since all pseudo-binary variables may be assigned any two values.

However, the choice of 0/1 is subjective. Any two different values are possible. The mathematical properties of ND will vary with the different choices, thus leading to clusters which depend of the way in which “presence” or “absence” is encoded.

# Previous Approaches

Most importantly, with this sort of scheme the pseudo-binary variables do no longer reflect the essence of the idea conveyed by the category.

The variable Place\_01, for instance, reflects the way the tuple is “affected” by belonging, say, to the categorical value “Center”, which is correct.

▪

# Previous Approaches

But now the original issue “How does the behavior of the individuals change according to their place of birth?” is replaced by “How does the behavior of the individuals change when their place of birth is ‘Center’?”

The two questions are not interchangeable.

# A New Approach (CENG)

In view of the limitations above, we aim at a coding scheme which allows us to replace the instances of the categories by numerical codes while preserving the original relationships determined by the original values of the categorical variables

# Basic Idea

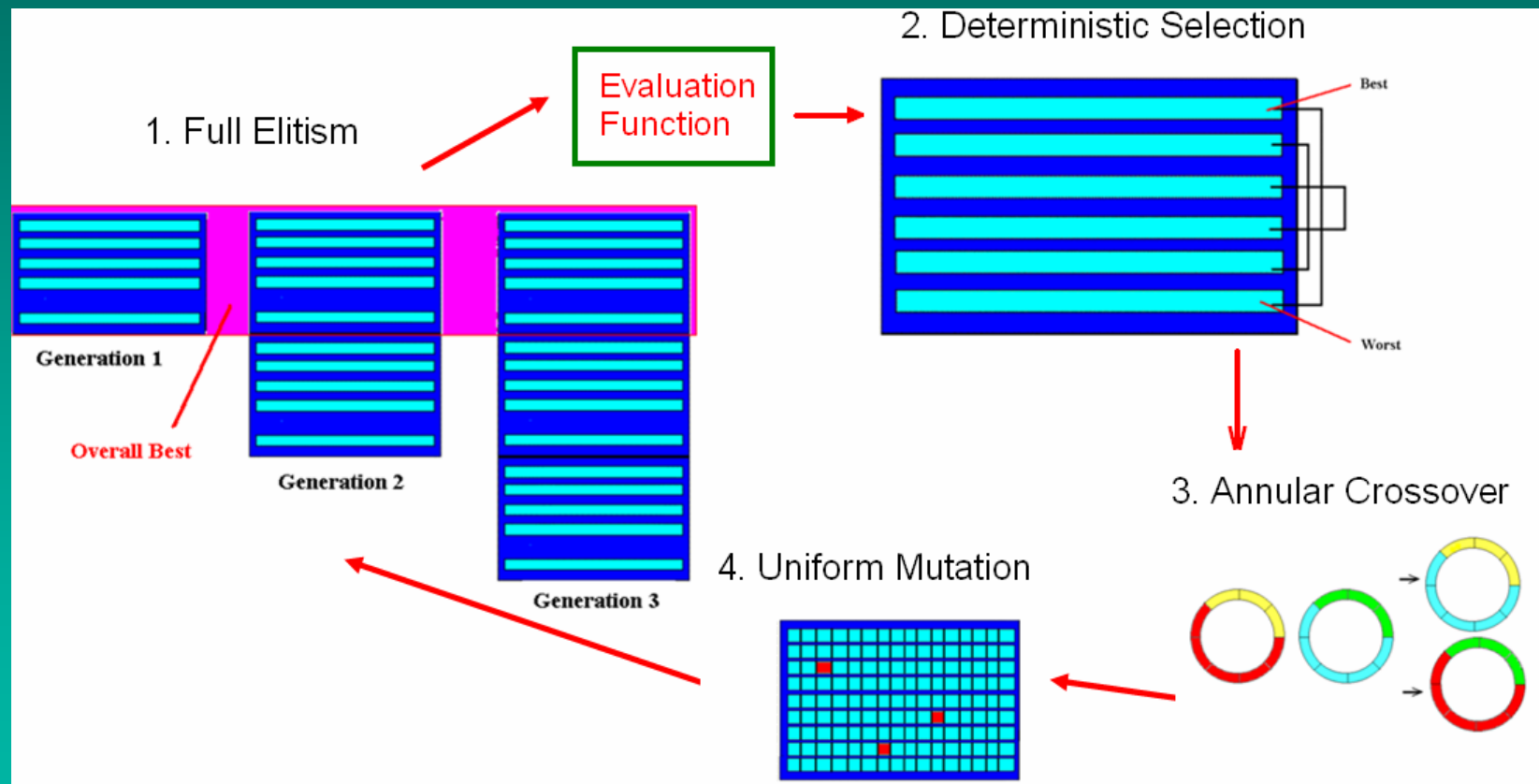
Obtain a set of codes which preserves all patterns embedded in the MDB by identifying, out of the potentially infinite alternatives, those which allow a best approximation of all combinations of multivariate functions.

# Basic Idea

Approximate the best code by using a genetic algorithm which extracts the code after training a set of neural networks.

The GA “proposes” a set of codes, the NNs are trained and return a worst case approximation error. This is minimized by the GA.

# Eclectic Genetic Algorithm



# Evaluation Function's Illustration

Assume a MD with 4 variables. Two numerical (N1, N2) and two categorical (C1, C2). C1 has two possible instances (I1, I2) and C2 has also two possible instances (I3, I4).

| N1     | N2     | C1 | C2 |
|--------|--------|----|----|
| 0.5527 | 0.3207 | I1 | I3 |
| 0.6955 | 0.4653 | I2 | I3 |
| 0.7495 | 0.4611 | I1 | I4 |
| 0.1077 | 0.5317 | I2 | I4 |
| 0.1493 | 0.5728 | I2 | I4 |
| 0.5146 | 0.9077 | I1 | I3 |
| 0.6456 | 0.8610 | I1 | I3 |
| 0.4420 | 0.8476 | I1 | I3 |

| N1     | N2     | C1 | C2 |
|--------|--------|----|----|
| 0.5527 | 0.3207 |    |    |
| 0.6955 | 0.4653 |    |    |
| 0.7495 | 0.4611 |    |    |
| 0.1077 | 0.5317 |    |    |
| 0.1493 | 0.5728 |    |    |
| 0.5146 | 0.9077 |    |    |
| 0.6456 | 0.8610 |    |    |
| 0.4420 | 0.8476 |    |    |



# Evaluation Function

- 1) The population consists of possible (binary encoded) numerical values for every tuple in the table.

| I1     | I2     | I3     | I4     |    |
|--------|--------|--------|--------|----|
| 0.8887 | 0.9329 | 0.4085 | 0.8392 | 1  |
| 0.1375 | 0.4726 | 0.1305 | 0.4281 | 2  |
| ...    | ...    | ...    | ...    |    |
| 0.0052 | 0.5651 | 0.6664 | 0.0658 | 70 |

Encoded in  
binary




# Evaluation Function


2) For every individual the codes are inserted into the categorical “area” of the MDB.

|        |        |        |        |
|--------|--------|--------|--------|
| 0.3411 | 0.6026 | 0.9958 | 0.6897 |
|--------|--------|--------|--------|

This is NDB



| N1     | N2     | C1 | C2 |
|--------|--------|----|----|
| 0.5527 | 0.3207 |    |    |
| 0.6955 | 0.4653 |    |    |
| 0.7495 | 0.4611 |    |    |
| 0.1077 | 0.5317 |    |    |
| 0.1493 | 0.5728 |    |    |
| 0.5146 | 0.9077 |    |    |
| 0.6456 | 0.8610 |    |    |
| 0.4420 | 0.8476 |    |    |

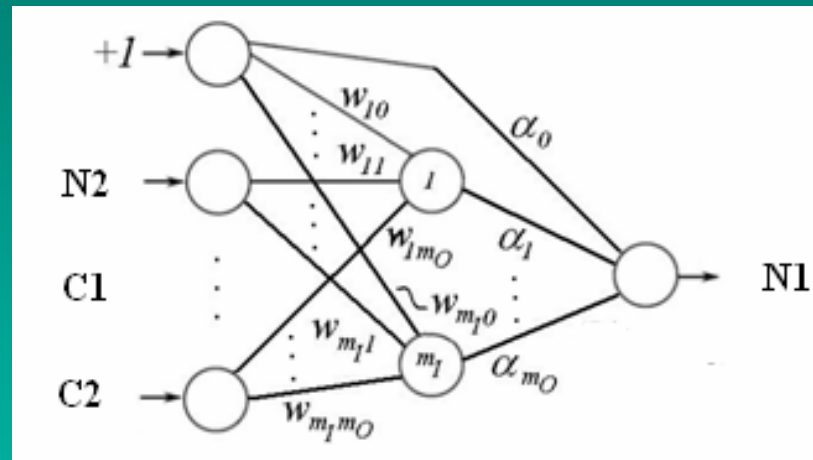


| N1     | N2     | C1     | C2     |
|--------|--------|--------|--------|
| 0.5527 | 0.3207 | 0.3411 | 0.9958 |
| 0.6955 | 0.4653 | 0.6026 | 0.6897 |
| 0.7495 | 0.4611 | 0.3411 | 0.9958 |
| 0.1077 | 0.5317 | 0.6026 | 0.6897 |
| 0.1493 | 0.5728 | 0.3411 | 0.9958 |
| 0.5146 | 0.9077 | 0.6026 | 0.9958 |
| 0.6456 | 0.8610 | 0.3411 | 0.9958 |
| 0.4420 | 0.8476 | 0.6026 | 0.6897 |

| N1     | N2     | C1     | C2     |
|--------|--------|--------|--------|
| 0.5527 | 0.3207 | 0.3411 | 0.9958 |
| 0.6955 | 0.4653 | 0.6026 | 0.6897 |
| 0.7495 | 0.4611 | 0.3411 | 0.9958 |
| 0.1077 | 0.5317 | 0.6026 | 0.6897 |
| 0.1493 | 0.5728 | 0.3411 | 0.9958 |
| 0.5146 | 0.9077 | 0.6026 | 0.9958 |
| 0.6456 | 0.8610 | 0.3411 | 0.9958 |
| 0.4420 | 0.8476 | 0.6026 | 0.6897 |

# Evaluation Function

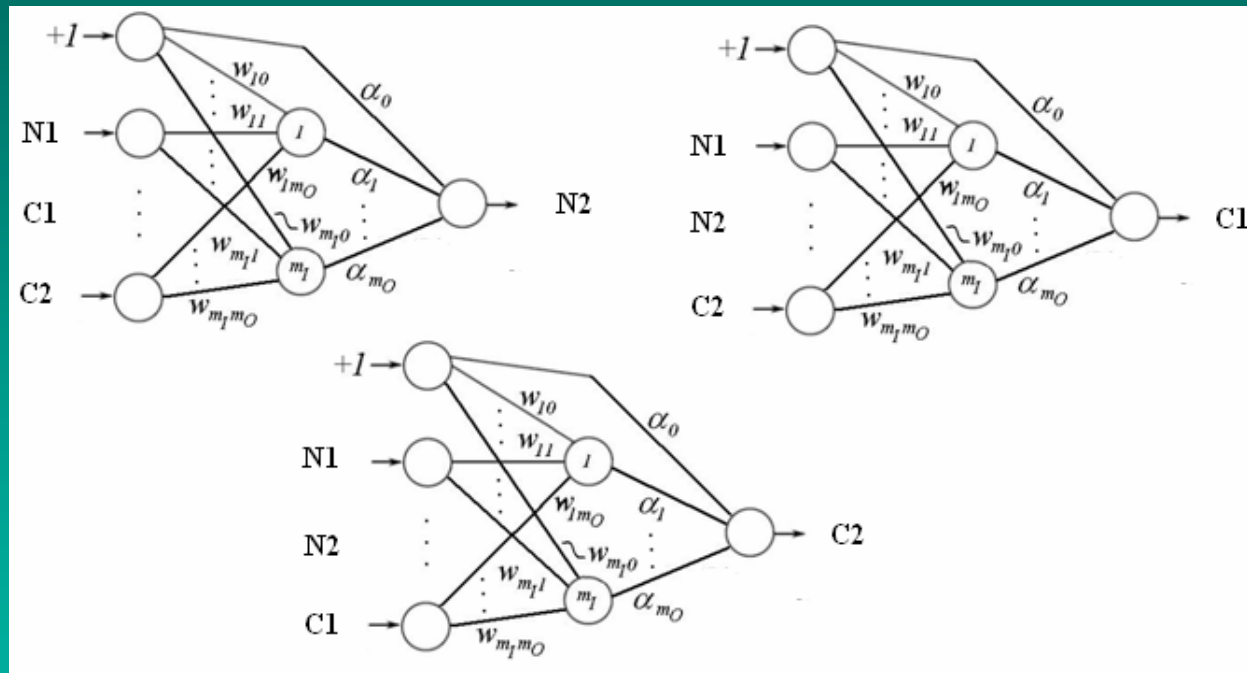
3) A Neural Network is trained where N1 is a variable dependent on N2, C1, C2.



The maximum absolute approximation error  $\varepsilon_1$  is recorded.

# Evaluation Function

The training process is repeated now making every variable a function of the others.



4)  $\text{Max}(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4)$  is the fitness of the individual of the GA.

# Resulting Codes

The GA is run for 500 generations or until the minimax approximation error is acceptable.

In the end the algorithm returns the set of codes which is able to express attribute  $i$  as a function of the remaining  $n-1$  AND this is true for all  $n$  attributes with the minimum possible error.

# Resulting Codes

*CENG* delivers the codes for every one of the *ct* instances of the categorical variables and *ND*: a version of *MD* in which all categorical instances are replaced by the proper numerical codes.

# Resulting Codes

| Age | Place of Birth | Years of Study | Race   | Sex | Salary    |
|-----|----------------|----------------|--------|-----|-----------|
| 55  | North          | 9              | White  | M   | 2,932.49  |
| 62  | North          | 7              | Asian  | F   | 23,453.37 |
| 57  | South          | 24             | Indian | F   | 1,628.61  |
| 56  | Center         | 18             | White  | M   | 4,069.62  |
| 49  | South          | 22             | Indian | F   | 3,650.23  |
| 27  | North          | 25             | Asian  | M   | 28,497.04 |
| 61  | North          | 19             | Other  | M   | 3,929.80  |
| 26  | North          | 11             | Indian | M   | 11,787.44 |
| 54  | Center         | 13             | Indian | F   | 625.69    |
| 58  | North          | 18             | Asian  | M   | 13,754.67 |

| Age    | Place of Birth | Years of Study | Race   | Sex    | Salary |
|--------|----------------|----------------|--------|--------|--------|
| 0.4928 | 0.1683         | 0.2000         | 0.1686 | 0.0221 | 0.0226 |
| 0.5942 | 0.1683         | 0.1200         | 0.3646 | 0.0231 | 0.1896 |
| 0.5217 | 0.2137         | 0.8000         | 0.2989 | 0.0231 | 0.0120 |
| 0.5072 | 0.0132         | 0.5600         | 0.1686 | 0.0221 | 0.0318 |
| 0.4058 | 0.2137         | 0.7200         | 0.2989 | 0.0231 | 0.0284 |
| 0.0870 | 0.1683         | 0.8400         | 0.3646 | 0.0221 | 0.2306 |
| 0.5797 | 0.1683         | 0.6000         | 0.7844 | 0.0221 | 0.0307 |
| 0.0725 | 0.1683         | 0.2800         | 0.2989 | 0.0221 | 0.0946 |
| 0.4783 | 0.0132         | 0.3600         | 0.2989 | 0.0231 | 0.0038 |
| 0.5362 | 0.1683         | 0.5600         | 0.3646 | 0.0221 | 0.1106 |

|                     |            |
|---------------------|------------|
| Fitness: .001281535 |            |
| North               | 0.16832423 |
| White               | 0.168648   |
| M                   | 0.02206635 |
| Asian               | 0.36462331 |
| F                   | 0.0230515  |
| South               | 0.21373081 |
| Indian              | 0.29890513 |
| Center              | 0.01316381 |
| Other               | 0.78435969 |

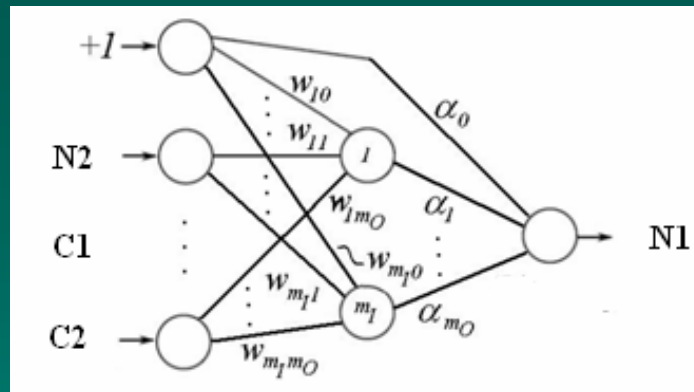
# Theoretical Justification

How do we know that CENG is robust?

Its feasibility rests on 2 proven facts:

- a) It is possible to approximate any experimental set of data as closely as desired using a 3 layer NN.
- b) Any function's global minimum may be found by an elitist genetic algorithm.





# Cybenko's Theorem

Let  $\varphi(\cdot)$  be a nonconstant, bounded, and monotonically-increasing continuous function. Let  $I_{m_O}$  denote the  $m_O$ -dimensional unit hypercube  $[0,1]$ . The space of continuous functions on  $I_{m_O}$  is denoted by  $C(I_{m_O})$ . Then, given any function  $f \in C(I_{m_O})$  and  $\varepsilon > 0$ , there exist an integer  $M$  and sets of real constants  $\alpha_i$ ,  $b_i$  and  $w_{ij}$ , where  $i=1, 2, \dots, m_I$  and  $j=1, 2, \dots, m_O$  such that we may define:

$$F(x_1, \dots, x_{m_O}) = \sum_{i=1}^{m_I} \left[ \alpha_i \cdot \varphi \left( \sum_{j=1}^{m_O} w_{ij} x_j + b_i \right) \right]$$

as an approximate realization of the function  $f(\cdot)$ , that is,

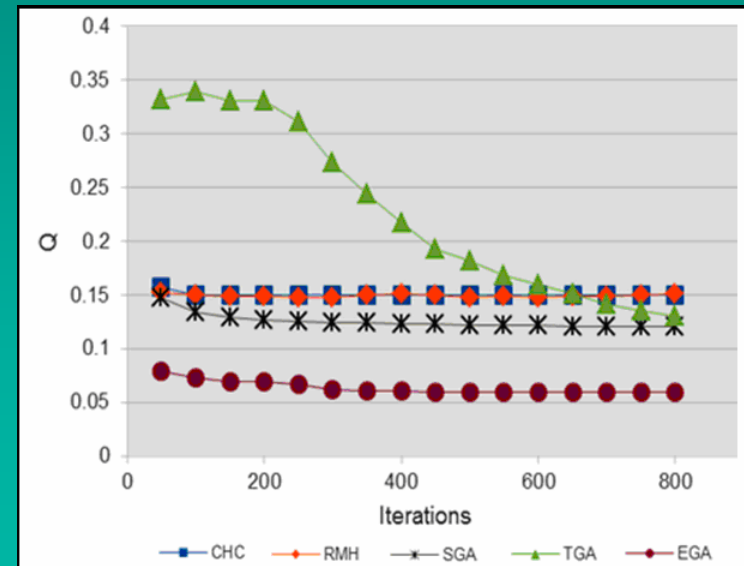
$$|F(x_1, \dots, x_{m_O}) - f(x_1, \dots, x_{m_O})| < \varepsilon$$

for all  $x_1, \dots, x_{m_O}$  in the input space

# Rudolph's Theorem

*The canonical GA with proportional selection, 1-point crossover and uniform mutation operators which maintains the best solution found over time after selection converges to the global optimum.*

However, we abandoned the CGA's paradigm in favor of a much more efficient GA found from an exhaustive statistical benchmarking process.



# Experimental Verification

The method relies on two stochastic methods.

The initial population of the GA is selected at random; the backpropagation training algorithm of the NN initializes the values of the network's weights randomly. Therefore, given two different seeds for the pseudo-random number generator we will get, in general, two different sets of codes.

However, **both sets will preserve the structure of the database (now numerical).**

# Experimental Verification

The following methods suggests itself.

- a) Run CENG starting with different seeds for the pseudo-random number generator.
- b) Use every run's result to find the clusters of the data.
- c) Compare the clusters.

If the clusters are similar, CENG has been shown to preserve the structures in the data, which was our goal to begin with.

# Experiments

We considered a mixed database consisting of 100 tuples and 6 attributes.

Three of the attributes were categorical and had 9 different instances, total.

We ran CENG with 4 different seeds: 31416, 27182, 61803 and 12357, giving origin to ND1, ND2, ND3 and ND4.

Next we clustered for 2,3,4,5 clusters for all NDs.

# Experiments

|          |       |        |        |        |
|----------|-------|--------|--------|--------|
| CLUSTERS | 2     | 27182  | 31416  | 61803  |
|          | 12357 | 69.00% | 77.00% | 80.00% |
|          | 27182 | X      | 90.00% | 77.00% |
|          | 31416 | X      | X      | 71.00% |
|          |       | AVG    |        | 77.33% |
|          | 3     | 27182  | 31416  | 61803  |
|          | 12357 | 68.00% | 70.67% | 76.00% |
|          | 27182 | X      | 92.67% | 79.33% |
|          | 31416 | X      | X      | 78.33% |
|          |       | AVG    |        | 77.50% |
|          | 4     | 27182  | 31416  | 61803  |
|          | 12357 | 78.25% | 78.25% | 80.00% |
|          | 27182 | X      | 96.00% | 81.25% |
|          | 31416 | X      | X      | 80.50% |
|          |       | AVG    |        | 82.38% |
|          | 5     | 27182  | 31416  | 61803  |
|          | 12357 | 82.80% | 81.80% | 84.80% |
|          | 27182 | X      | 93.60% | 84.60% |
|          | 31416 | X      | X      | 85.40% |
|          |       | AVG    |        | 85.50% |

The result of comparing the clusters for the different codes' sets is shown on the left.

The best case corresponds to a 96% coincidence, which is remarkable.

The worst case corresponds to 68% coincidences (for 3 clusters).

# Experiments

The number of possible clustering configurations for 3 clusters and 100 tuples is  $3^{100}$ .

The probability of correctly picking the right configuration 68% of the time is  $\alpha = (1/3)^{68} \approx 3.6e-33$ .

On the other hand, there are  $\beta = C(100,68) = 100!/(32!68!) \approx 1.4e+26$  ways to attempt 68 correct guesses.

Hence, the probability of correctly achieving a score of 68% by chance alone is  $\alpha \times \beta \approx 5e-7$ : 50 in a million!

In the best case result, where 96% of the clusters matched, a similar calculation yields a probability of  $3.6e-39$  of this event to happen by chance.

# Conclusions (1)

We have described how to target on a set of codes which allows us to preserve the patterns embedded in a mixed database.

This is theoretically possible because both the learning problem (via NNs) and the optimization problem (via GAs) have been show to converge to optimum values.



## Conclusions (2)

We have also solved the underlying multi-objective optimization problem by optimizing the minimax norm.

We argued that, given the stochastic nature of the GAs and the NNs, running CENG with two roots of the random number generator will result in different sets of codes. Both, however, should preserve the structure of the ND.

## Conclusions (3)

Given the last point, it is reasonable to assume that clusters found starting with one set of codes and another derived from a different seed will behave similarly.

Accordingly, we trained for 2,3,4 and 5 clusters the same data set.

Even in the worst experimental case (68% matching) the odds against it being due to chance are very small.

# Limitations

- The codes are dependent on the data. That is, every database has to be trained and the codes thusly found are, in general, not applicable to other databases.
- The method is computationally intensive.

Both disadvantages are minor when considering the advantages obtained.

# Possible Applications

- Generalization of numerical clustering algorithms
- Unbiased data mining of categorical data bases
- Data mining of unstructured data
  - Texts
  - Images
  - Audio