PhUSE 2014

Paper AD05

Exploring Safety and Diagnostic Data using Self-organizing Maps

Colin de Klerk
inVentiv Health Germany GmbH, Eltville am Rhein, Germany

ABSTRACT

Using self-organising map (SOM) neural networks can reveal features and clusters of similar characteristics in the underlying data. In a SOM-based visualisation, multidimensional data are mapped onto a 2D grid and can help to identify initial regions of interest. Subsequently these can be examined more closely by conventional means.

While SOM techniques are already integrated into SAS® Enterprise Miner, we apply them by first principles to sampled safety and diagnostic data using Base SAS® and SAS®/GRAPH. Nominal, categorical input data present a particular challenge, since SOM models typically assume simple, continuous input data.

The results reveal certain predictable relationships, but also less intuitive patterns that provide potentially interesting kernels on which to concentrate further analysis.

INTRODUCTION

The self-organising or feature map was introduced by Kohonen [1] in the latter decades of the twentieth century as a type of artificial neural network that can project higher-dimensional input data onto, say, a 2D grid. Related features in the higher-dimensional data are generally preserved in the topological arrangement of the projection, which means that inputs containing similar features tend to be reflected in nearby nodes in the grid. Thus a SOM, also known as a feature map, can reveal clusters or other artefacts that are not obvious in the input data.

The basic operation of a SOM involves iteratively locating the best matching unit (BMU), i.e. the node in its 2D grid that is “closest” in some deterministic sense to a specific item in the input data. Closeness in this context is measured by a distance metric appropriate to the application.

Unlike many neural networks that require supervision, or explicit intervention to associate training data with known outputs, a SOM is tuned in an unsupervised fashion by a process of competitive learning. During the training phase, the reference vector of the BMU itself as well as those of a number of its immediate neighbours are then adjusted so that they more closely resemble the training item. The system eventually reaches a stable equilibrium state when the adjustment amounts become vanishingly small.

Once trained, a SOM can be used as a classifier to determine groups of similar multidimensional inputs. We elaborate on this in the discussion of the results.

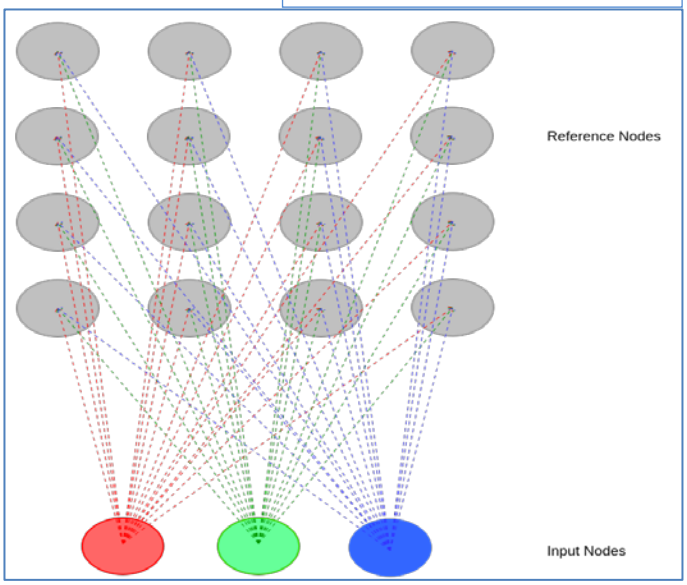
APPROACH

In this paper, we briefly discuss the theoretical underpinnings. Then we demonstrate a simple SOM operating on bounded, continuous data, followed by a more complex example that examines safety data from a series of clinical trials to investigate hidden relationships in the data.

The clinical data contain nominal or categorical data which require special consideration. Two approaches on treating such data will be discussed.

THE SELF-ORGANISING OR FEATURE MAP

Figure 1: A Self-organising Map



The self-organising map is a single-layer structure in which each input is fully connected with each node of the grid. Grid nodes are independent of one another and not connected directly. The number of nodes and their placement within the grid are determined by the application.

Within the 2-dimensional feature map, nodes are typically arranged in hexagonal or rectangular configuration, whichever is more suitable to display that particular map. Every non-border node is aligned with 6 neighbours in the hexagonal configuration and with 4 in the rectangular. In this paper, we use the rectangular option.

Every input and each node in the SOM can be represented by vectors with the number of components corresponding to the number of inputs. While there is a trade-off between the total number of nodes and the computational resources required to calculate the states of the system, the dimensionality of the system and thus the size of the vectors appears to be of secondary importance.

TRAINING THE SOM

Paraphrasing Kaski [3], competitive learning is an adaptive process in which certain neurons in an artificial neural network gradually become sensitised to different feature categories via samples from the input data. Neurons come to recognise input features competitively: the neuron that is best able to represent an input X wins the competition.

Put simply, the best matching unit is the neuron with the minimum distance between its reference vector $\mathbf{m}_c$ and the input vector **x.** The distance is often Euclidean, i.e. $d(x, m_i) = \sqrt{(x_i - m_{i1})^2 + \cdots + (x_j - m_{ij})^2}$ for $\mathbf{m}_i$, the reference vector with j components of the i-th neuron. For an input X, the index of C, its BMU is given by $c(x) = \underset{i}{\operatorname{argmin}}\{\|x - m_i\|^2\}$.

Once isolated, the BMU forms the kernel of a local group of neighbouring neurons for which the vector components or weights are adapted to match the input even better.

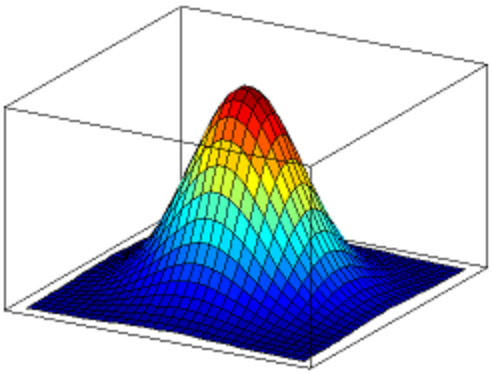


Figure 2: Degree of adaptation within the neighbourhood of the BMU (http://www.cs.hmc.edu/~kpang/nn/5.bmp)

The degree of adaptation is based on the distance d(**x**, $\mathbf{m}_c$). It is greatest for the BMU itself as represented by the dark red peak in the figure above and then drops rapidly to nothing (blue) beyond the periphery as measured by a (typically Gaussian) neighbourhood function, $h_{ci} = h(\|r_c - r_i\|)$ where $\mathbf{r}_c$ and $\mathbf{r}_i$ are the *position* vectors of the BMU≡C and another node with index i within the grid.

In practice, the radius that is searched to find the BMU is initially very large - as much as half of the total size of the grid. This allows the search for the BMU to start out globally over the entire grid. Both the search and the neighbourhood radii drop off over time to fine-tune locally as the learning proceeds.

Putting all of this together allows us to express an adaptation rule for the reference vectors of the SOM grid as the system progresses through the training iterations. For two subsequent iterations t and t+1, the change in the reference vector $\mathbf{m_i}$ can be expressed by $m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)]$.

Once a SOM has been trained, it can be used as a classifier to group the input data. We can introduce further examples to the system and they will be allocated to the most suitable node of the SOM and hence the group that this node represents.

AN EXAMPLE OF A SIMPLE SOM

RGB values are one of many ways that colours can be encoded in computer systems. A particular colour is represented by 3 components corresponding to the amounts of each of the primary colours red, green and blue, usually in the range [0,255] although the range can also be normalised to, say [0,1]. RGB values are often expressed compactly in hexadecimal notation, e.g. #FF0000 is “red” and #0000FF is “blue”.

If we set up a rectangular SOM with m rows of n columns, depicting the state of each node is a trivial exercise of displaying a symbol of the appropriate RGB colour value at the location of each node in the grid. This proved to be difficult using the GPLOT or even SGPLOT procedures in SAS 9.2 because of the large number of colour values in the grid. The implementation was thus done using the GANNO procedure.

1. We began by initialising the system so that each of the m=20 x n=20 nodes had a reference vector of size 3 containing normalised, normally distributed random values between 0 and 1. Note the paucity of extreme values such as black (#000000) and white (#FFFFFF).

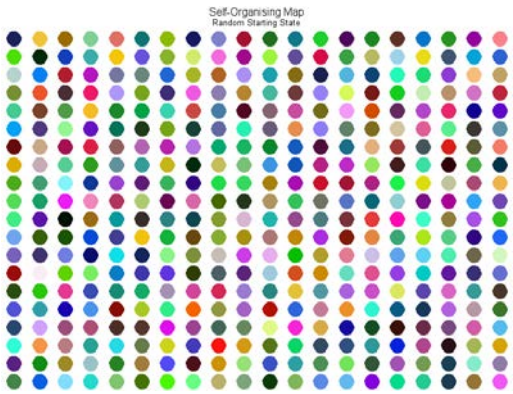


Figure 3: Initial, random state

2. We then presented the system with a training set of 10 known colour vectors: black, red, green, dark green, blue, dark blue, yellow, orange, magenta and white 

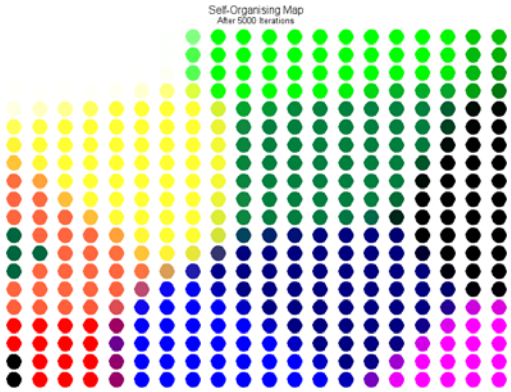


Figure 4: After 5000 iterations

3. After 5000 iterations of the learning algorithm, the system attained the stable state shown in Figure 4.

Given that as a rough guideline for the number of iterations required, Kohonen recommended around 500 times the number of nodes in the grid [1, p1469], which for this system would be around 200000, it is clear that this system has already delivered an acceptable result in much less than that.

THE GENERALISED SELF-ORGANISING MAP

THE TROUBLE WITH CATEGORICAL DATA

Depending on what distance metric is chosen, a SOM works best with continuous, real-valued data as in the RGB example, or even with discrete data because these values can be *simply ordered* (e.g. 1.3 < 2.5 < 6.7) and adjusted by arbitrary amounts. However, data records in many real-life applications are frequently nominal or categorical (e.g. 1=Male, 2=Female) in nature where the coding is arbitrary and any ordering is entirely specious (1 is less than 2, certainly, but in what uncontroversial sense is Male less than Female?). Note that some categorical data can indeed be ordered *semantically*, e.g. severity options ("mild", "moderate" and "severe"), or alcohol use ("light", "heavy").

Among the approaches that have been proposed to deal with this problem, the "1-of-k" method is well known. There are many variants, one of which involves replacing a single column (Code) that can hold any one of the available code values with a group of columns corresponding to each available option that can only hold binary values (0 or 1).

As an example, consider 2 input values x_1 = (…, 3, …) and x_2 = (…,996, …) with the values of the category Race (Table 1) being 3="Asian" and 996="Not reported" respectively. We drop the existing column and replace it with 6 new columns, each corresponding to one of the options. We fill the new columns with 0 everywhere except for the row corresponding to the code value, which is set to one. Thus we obtain new vectors with 5 additional components x'_1 = (…, 0,0,1,0,0,0, …) and x'_2 = (…,0,0,0,0,1,0, …).

Code	Decode	W	B	A	AI	NR	M
1	White	1	0	0	0	0	0
2	Black or African American	0	1	0	0	0	0
3	Asian	0	0	1	0	0	0
4	American Indian or Alaska Native	0	0	0	1	0	0
996	Not Reported	0	0	0	0	1	0
999	Multiple	0	0	0	0	0	1

Table 1: 1-of-6 conversion

The 1 of k method suffers from two main disadvantages. Apart from increasing the size of the vectors representing the inputs and nodes and thus the computational resources required by the system, as Hsu observes [2], the resulting feature maps do not topologically preserve the features of the input data (clustering is impaired or even disabled), so we lose much of the power of the feature map.

THE DATA HIERARCHY APPROACH

Hsu describes a method [2] to impose a hierarchy onto the category options so that real-number ordering and adjustment becomes possible. The idea is that the distance between two semantically closer options is lower than that between two less-related options.

Looking at the example in the Figure 5, we note intuitively that "Dose reduced" is closer semantically to "Dose not changed" than it is to "Dose increased". This is reflected in the sum of the weights (1+0.5) if we traverse the edges from "Dose reduced" to "Dose not changed" within the tree. Similarly, the distance (1+0.5) from "Dose increased" to "Dose not changed" is less than the distance from "Dose reduced" to "Dose increased" (1+1) and much less than the distance from "Drug withdrawn" to "Unknown" (1+1+2+1).

Data hierarchies (DH) are tree structures with the category options as leaf nodes (a *leaf* node is distinct from an *internal* node because it has no descendents). DHs should ideally be carefully arranged by experts in the application domain; this is merely simulated in this paper. Within the DH, any category option and indeed *any arbitrary point* can then be mapped to a vector with two components: The first is the text name of the leaf node corresponding to the category option (e.g. "Drug Withdrawn"). The second is a real number that ranges from 0 at the root to the cumulative weight of the edges connecting the root to that leaf node (2+1 for "Unknown"). For example, P ("Drug Withdrawn", 0.75) and Q ("Unknown", 2.2) are 2 arbitrary points positioned within the tree as shown in Figure 5. The distance between them is 2.95.

An essential concept in calculating the distance between points in a DH is the lowest common ancestor (LCA). This is the lowest-level node that is common to both. Thus "?1" is the LCA for "Drug Withdrawn" and "Drug Interrupted", "?0" (the root node) is the LCA for points P and Q and Q itself is the LCA for Q and "Unknown".

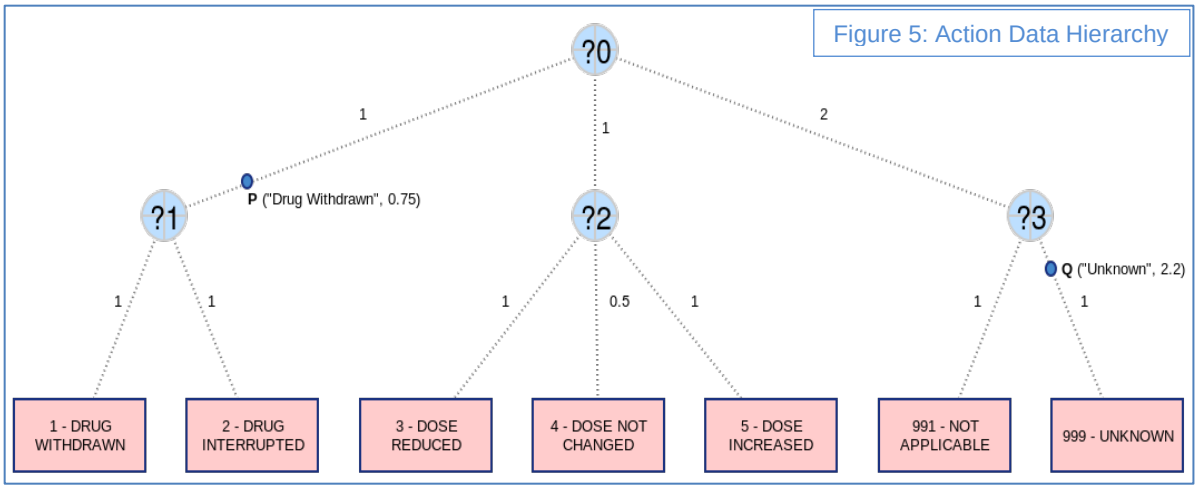


Figure 5: Action Data Hierarchy

Within a data hierarchy, we can find the distances between categorical items. As we know, the BMU is always the node in the SOM with the lowest aggregate distance to an input vector. During the training phase of the SOM, the BMU and its neighbours will be adjusted to make the distance even smaller. Hence we also need to be able to adjust the distances of categorical items. We do this by repositioning their representing points in the DH with the side effect that the anchoring leaf may change as a result.

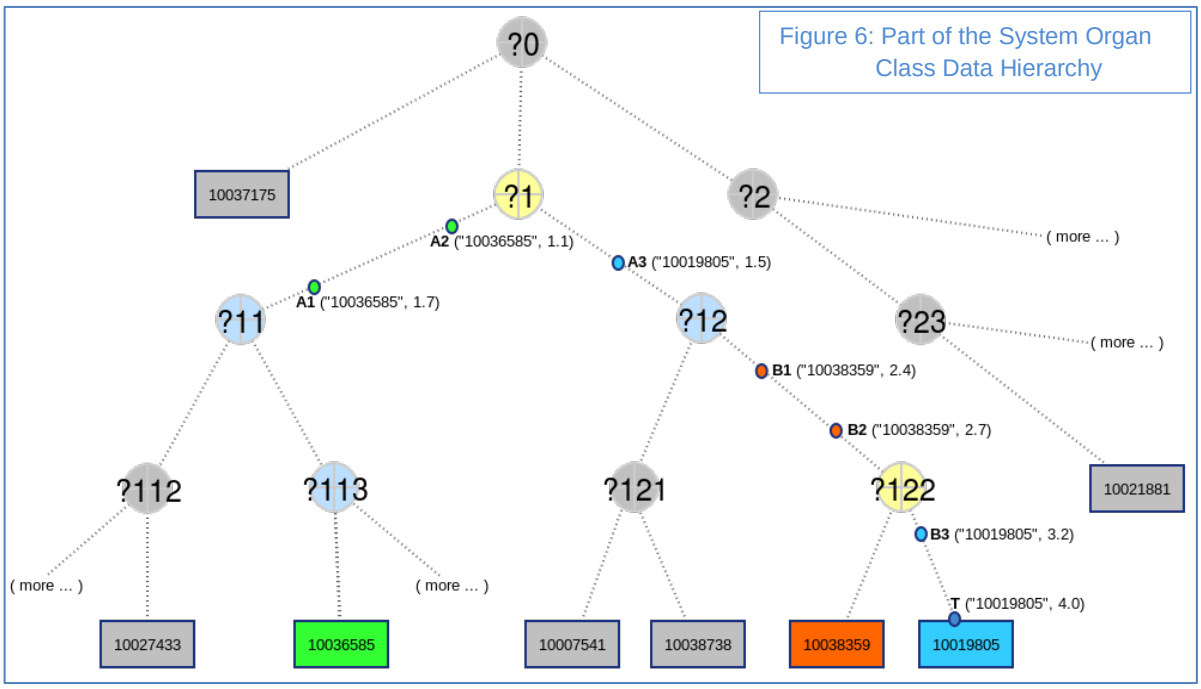


Figure 6: Part of the System Organ Class Data Hierarchy

We discuss the approach in reference to Figure 6, which shows part of the System Organ Class data hierarchy and consider A_i and B_j, arbitrary points in the DH. We will examine 4 different cases. Recall that any point in a DH has 2 components: the "anchoring" leaf and the cumulative weight from the root to this leaf.

Assume that we wish to adjust the A_i and B_j so that they move in the direction of target point T("10019805", 4.0) which is at the blue leaf node "10019805".

1. The starting point A1("10036585",1.7) has its leaf at the green node "10036585". The LCA between the green and the blue nodes is "?1" which means that the A1 is below the LCA. The total distance between A1 and T is found by traversing the tree from A1 to T as 0.7 + 1.0 + 1.0 + 1.0 = 3.7. Assume that the adjustment amount δ=0.6 which moves us after adjustment to the point A2("10036585",1.1). This point is also below the LCA and thus has the same leaf node as A1.
2. The point A2("10036585",1.1) also has the green node as its leaf and is also below the LCA "?1". The total distance to T is 3.1. A δ of 0.6 of is sufficient to push us across the LCA after adjustment to the branch that

leads to T. So not only does the numerical part change, but we also change the leaf node to that of the target and move to the point A3("10019805",1.5).

3. The starting point is B1 ("10038359",2,4), which has the orange node "10038359" as its leaf and is 2.6 units away from T. The LCA between this leaf and the target leaf is "?122" which puts B1 above the LCA. An adjustment of δ=0.3 does not result in crossing the LCA, so the leaf stays the same and the end point after adjustment is B2 ("10038359",2,7).
4. The point B2 ("10038359",2,7) also has the orange node "10038359" as its leaf and is also above the LCA "?122". The total distance to T is 1.3. A δ of 0.5 is sufficient to push us across the LCA after adjustment to the branch that leads to T. So not only does the numerical part change, but we also change the leaf node to that of the target and move to the point B3 ("10019805",3.2).

PROGRAM IMPLEMENTATION

We implemented a GSOM in SAS 9.2 to process around half of our approximately 1600 adverse event and demographic data records derived from real trial data. We used the MedDRA System Organ Class as an identifier for the event. In the interest of producing legible graphics, we created a SOM grid of just 5 x 5 nodes. A really useful SOM would probably be much larger. We set the system up to iterate 7500 times.

ASSUMPTIONS

The largest data hierarchy was the System Organ Class with a depth of at most 5 from root to leaf. Thus the maximum distance between any 2 nodes was around 10. To keep the influence of all variables approximately equal, we therefore scaled all of the numerical values (e.g. age, height and weight) to the range [0,10].

DISCUSSION OF INPUT DATA

We selected the 5 numeric fields **Age, Base Height, Base Weight, Base BMI** and **Smoking Dose**. Clearly the weight, height and BMI are related to one another, but the other variables are fairly independent. We also created data hierarchies for these 9 categorical items:

Action Taken = {Drug Withdrawn, Drug Interrupted, Dose Reduced, Dose Not Changed, Dose Increased, N/A, Unknown}
Alcohol Use = {Abstinent, Light, Moderate, Heavy}
System Organ Class = <<a subset of 26 items from the MedDRA dictionary>>
AE Outcome = {Recovered, Recovering, Recovered with Sequelae, Not Recovered, Fatal, Unknown}
Race = {White, Black or African American, Asian, American Indian or Alaska Native, Not Reported, Multiple}
AE Severity = {Mild, Moderate, Severe}
Sex = {Male, Female}
Related = {No, Yes}
Serious = {No, Yes}.

TECHNICAL DETAILS OF PROGRAM

To attain acceptable run times, we used SAS hash objects for look ups and created user-defined functions via the FCMP procedure. A full 7500-iteration run with some post-production to generate the graphics took well under a minute to run.

RESULTS

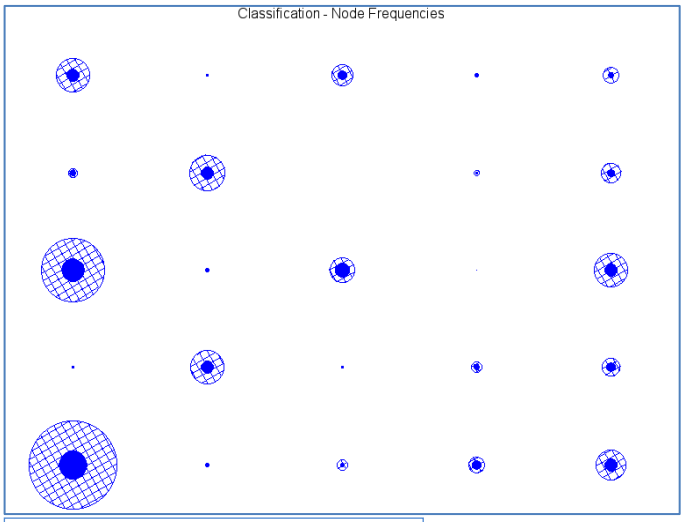


Figure 7: Classification Results

Figure 7 shows the results of the trained SOM operating as a classifier on the non-training half of our 1600 records. Each node represents an artificial record with certain characteristics acquired during the training phase.

The sizes of the outer circles represent the relative frquencies with which that node was found to be the BMU for an input record. The node at the bottom left was the BMU most often, while the one at centre-right was very rarely selected.

The inner circles represent the degree of fit between the reference and the input vectors. The smaller the size of the inner circle relative to its outer circle, the lower the distance and thus the better the fit.

The next few figures show some of the characteristics of the various nodes. Figure 8 shows the numeric subject features age (red), BMI (blue) and tobacco use (green). We can see that age is greatest at the bottom and left, BMI is greatest at the top right and tobacco use at centre bottom. Since these are numerical variables, the size of the wedge is a direct indicator of the magnitude of the factor.

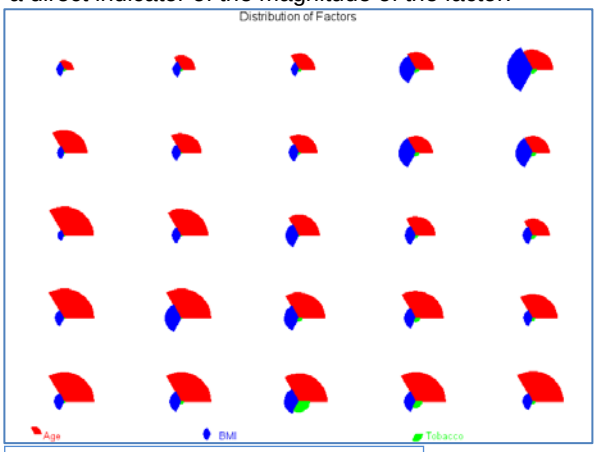


Figure 8: Age, BMI, Tobacco Use

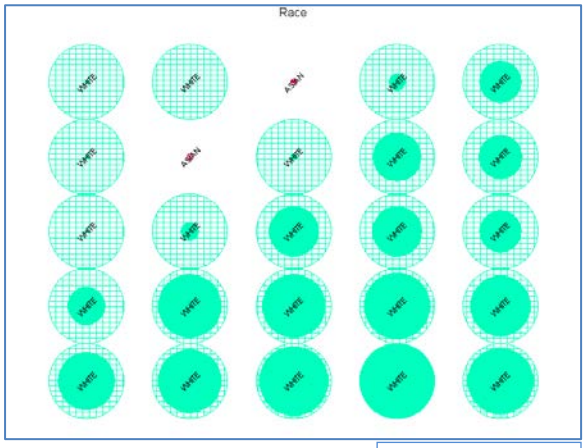


Figure 9: Race

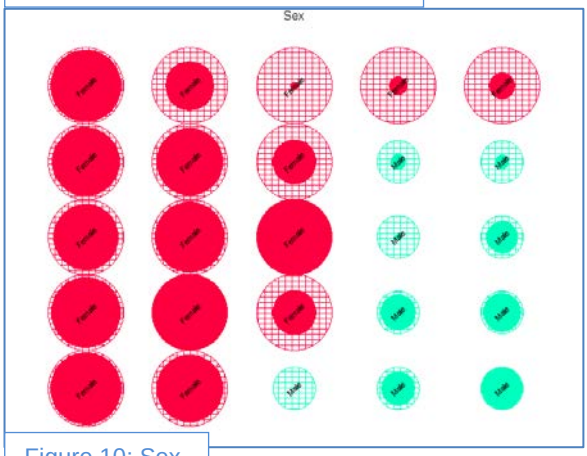


Figure 10: Sex

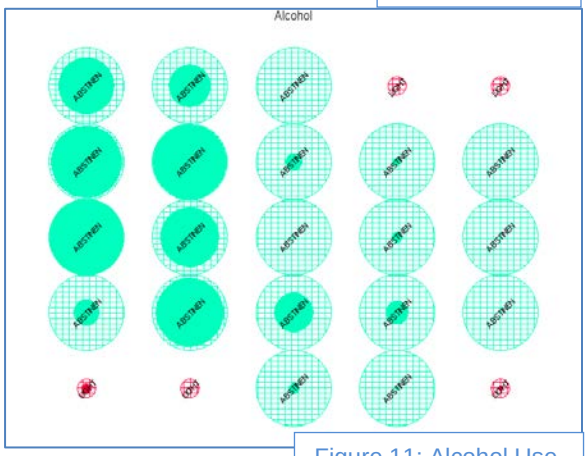


Figure 11: Alcohol Use

In Figures 9 through 13, which show a few of the categorical variables, the sizes of the outer circles reflect the frequencies of the options in the node data and are the same size for all nodes mapped to that leaf. The smaller, solid circles represent the numerical second component of the mappings in the data hierarchy for that node. A leaf is totally solid, the closer the point is to the root, the less solid it appears.

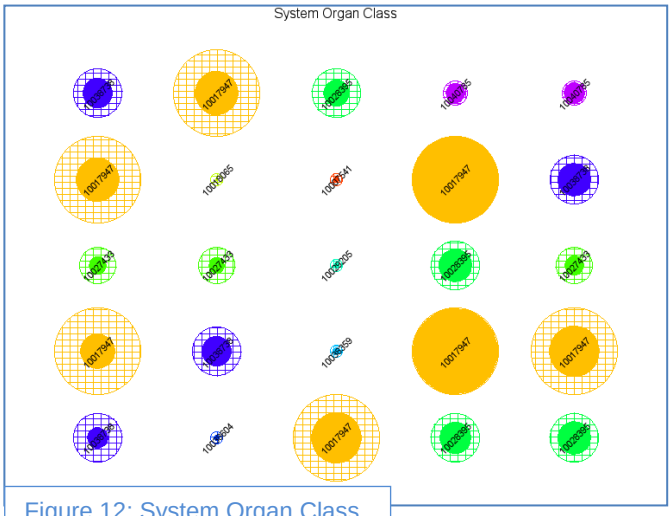


Figure 12: System Organ Class

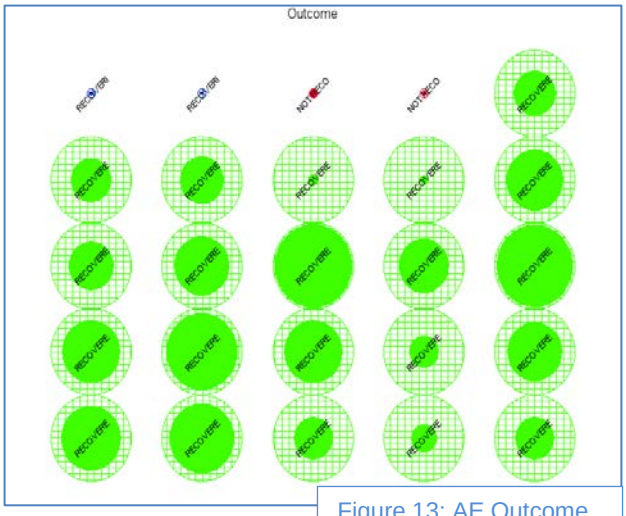


Figure 13: AE Outcome

Note that Figures 8 to 11 show subject characteristics while Figures 12 and 13 show features of the adverse events. Clearly we can describe the defining features of any node either graphically or by other means (e.g. we state that all of the nodes in the trained SOM had the severity value “mild”, were not serious and were not related to the study drug).

INTERPRETATION OF RESULTS

A trained SOM can be used as a classifier to sort and group the data. If we introduce fresh inputs to the system, each will each be allocated to the most suitable node of the SOM, the BMU for that input. We can be sure that the inputs are more similar to their BMU and to other inputs already allocated to this BMU than they are to any other node. Thus we can group the inputs, possibly in quite subtle ways because the *total* distance across all of the component variables making up the input and reference vectors is taken into account – in essence, we perform *whole-vector* queries.

Combining the classification shown in Figure 7 with the characteristics shown in Figures 8 to 13 or described above, we can state, for example, that the most frequent BMU node at the bottom left describes a fictitious adverse event experienced by an older, non-smoking, white female with moderate BMI who is light drinker of alcohol. This subject fully recovered from a mild, non-serious event with the SOC 10038738 (Respiratory, thoracic and mediastinal disorders) that was not considered to be related to the study medication.

By careful selection of training sets, we can adjust the results as needed. Alternatively, we can create queries by given nodes specific values – they will then attract items in the input data that quite closely share these values – but here we must be careful to control the side effects of the other attributes.

We can use the results to decide where we would like to concentrate limited resources for more stringent analyses. Thus from Figure 7, we can see immediately that there appears to be a skewing or bias in the data by the large node at the bottom left. Similarly, we might be inclined to examine the much smaller groups of events represented by the smaller circles – they may reveal rare events that might merit further examination.

It was stated that the data hierarchies should be set up by experts. In this paper, we created fairly arbitrary hierarchies (e.g. the SOC weights were mostly 1 although there were different branch heights) and much could be done to improve upon them. The clustering ability of the SOM would benefit greatly from more careful weightings of the edges.

In addition, the selection of fields was entirely arbitrary and thus the results are not as targeted as they would be in a more realistic analysis in which it would be essential to select appropriate (risk) factors for a specific application if the system is to have any useful predictive value. We implemented the program to be as flexible as possible and indeed adding a new variable is a fairly trivial exercise, with the proviso that a suitable data hierarchy is available for categorical data.

CONCLUSION

We have shown how a SOM can be set up and trained by introducing data to it and allowing the reference vectors for the nodes to be adjusted to minimise the distance between the reference vectors and the inputs. This trained SOM was then used to classify the input data in an initial exploration step.

Once the SOM has been trained, we obtain a reproducible classifier that allows us to distinguish adverse events that are more alike in their entirety – i.e. over all of the numeric and categorical variables considered - rather than on just one or two criteria that superficially resemble one another. This could be useful in selecting contrastive analysis populations or verifying hypotheses prior to more rigorous analysis by other means. Thus the SOM serves more to explore the data than as a formal analysis tool and is indeed used in that way in the fields of data mining.

GLOSSARY

LCA – Lowest Common Ancestor

RGB – Red-Green-Blue coding for colour

Artificial Neural Network: In machine learning and related fields, artificial neural networks (ANNs) are computational models inspired by an animal's central nervous system (in particular the brain) which is capable of machine learning as well as pattern recognition. Artificial neural networks are generally presented as systems of interconnected "neurons" which can compute values from inputs.

REFERENCES

[1] Kohonen, Teuvo, “The Self-Organizing Map”, Proceedings of the IEEE, Vol. 78, No. 9, September 1990

[2] Hsu, Chung-Chian, “Generalized Self-Organizing Map for Categorical Data”, IEEE Transactions on Neural Networks, Vol. 17, No. 2, March 2006

[3] Kaski, S., Data exploration using self-organizing maps. Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series No. 82, Espoo 1997, 57 pp. Published by the Finnish Academy of Technology. ISBN 952-5148-13-0. ISSN 1238-9803. UDC 681.327.12:159.95:519.2

RECOMMENDED READING

1. http://en.wikipedia.org/wiki/Self-organizing_map
2. http://www.ai-junkie.com/ann/som/som2.html

CONTACT INFORMATION

Colin de Klerk
inVentiv Health Germany GmbH
Grosse Hub 10d
65344 Eltville-am-Rhein
Tel: +49 (0) 6123 70437 14
colin.deklerk@inventivhealth.com

>> TRANSFORMING PROMISING IDEAS
INTO COMMERCIAL REALITY

Join us on Facebook and Twitter