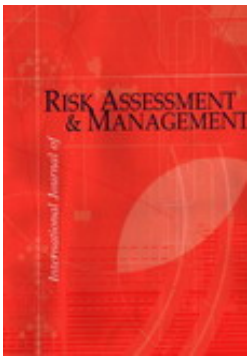


International Journal of Risk Assessment and Management

ISSN online: 1741-5241 - ISSN print: 1466-8297
https://www.inderscience.com/ijram

Rule based design using clustering for knowledge acquisition

Peter Grabusts

DOI: 10.1504/IJRAM.2022.10053794

Article History:

Received:	20 July 2020
Accepted:	18 January 2021
Published online:	02 February 2023

Rule based design using clustering for knowledge acquisition

Peter Grabusts

Faculty of Engineering,
Rezekne Academy of Technologies,
Atbrivoshanas Alley 115, Rezekne, LV-4601, Latvia
Email: peteris.grabusts@rta.lv

Abstract: Data analysis can be done by expert system decisions on system status according to system input and output data. For the purpose of data analysis, there is often a need to classify data or to find regularities therein. The results of the regularity search can be expressed by the IF-THEN production rules. The use of different approaches – with clustering algorithms, neural networks – makes it possible to obtain rules that characterise data. Knowledge acquisition in this paper is the process of extracting knowledge from numerical data in form of rules. Rules acquisition in this context is based on clustering methods. With the help of the K-means clustering algorithm, rules are derived from trained neural networks. The rule-making methodology is demonstrated on a sample basis of IRIS data. The effectiveness of the obtained rules is evaluated.

Keywords: data analysis; decision making; clustering algorithms; K-means; neural networks; risk analysis; system modelling; rule acquisition; rule base.

Reference to this paper should be made as follows: Grabusts, P. (2022) 'Rule based design using clustering for knowledge acquisition', *Int. J. Risk Assessment and Management*, Vol. 25, Nos. 1/2, pp.21–30.

Biographical notes: Peter Grabusts received his Dr. Sc. Ing. degree in Information Technology from Riga Technical University in 2006. Since 1996, he has been working at Rezekne Academy of Technologies. Since 2014, he has been a Professor at the Engineering Faculty. His research interests include data mining technologies, neural networks and clustering methods.

This paper is a revised and expanded version of a paper entitled 'Extraction rules from trained RBF neural networks' presented at *Proceedings of the 5th International Conference "Environment. Technology. Resources"*, Rezekne, Latvia, 16–18 June, 2005, Vol. 1, pp.33–39, DOI: https://doi.org/10.17770/etr2005vol1.2128.

1 Introduction

The purpose of intelligent data analysis (IDA) is to find hidden regularities in the data and express them by rules. The field of IDA is considered very important due to the definition of data as knowledge. IDA can also be described as a process that "looks for

previously unknown, non-trivial, interpretable regularities in decision-making in various applications of human activity" (Tan et al., 2019; Fayyad et al., 1996).

Different models of knowledge representation are known. They can be divided into the following classes: logical models, semantic networks, frames, production rules model. In the following, only production rules model will be observed, as it is based on the notional rules that allow you to assign knowledge in the form of IF-THEN, that is to say the general form of production rules:

$$IF\left\{\left(Event\,1\right)AND\left(Event\,2\right)AND\ ...\left(Event\,N\right)\right\}THEN\ldots$$

The advantage of production rules systems lies in the fact that their conclusions are essentially similar to those of experts. The motivation is following:

- finding previously unknown regularities in the data
- ability to express the found regularities in a user-friendly and understandable way.

The author's scientific interests are related to artificial neural networks and clustering, and the choice of rule-making method based on the radial-basic function network (RBF) is justified, because this neural network uses clustering in the training phase.

With the growing interest in the IDA field, studies on the possibilities of processing IF-THEN rules by various methods have become noteworthy. This can provide the knowledge base model that underpins further data analysis and risk management.

The aim of this work is to investigate task classes that implement regularisation and rule making methods from numerical data through clustering for risk analysis.

2 Clustering in the neural networks

2.1 The suitability of neural networks for rule acquisition

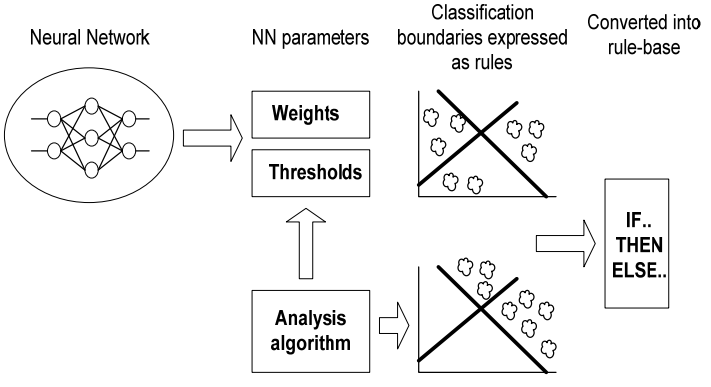
The use of neural network technologies can extend the capabilities of neural networks in data analysis (Fausett, 1993). There are many studies that allow reading linguistic information from artificial neural networks (Craven and Shavlik, 1995; Andrews et al., 1995; Dutch et al., 2004). In general, neural networks have a very good ability to impart 'empirical knowledge' in the form of input data, but information on network performance is predominantly expressed by a trained neural network structure and weights, which makes it difficult for the user to interpret the results obtained. It can be said that the operation of a neural network is similar to that of a "black box", with no explanation of its operation. This partially limits the use of neural networks in many applications where knowledge acquisition through subsequent decision making is essential.

Rule extraction process in a common case is shown in Figure 1 (Andrews and Gewa, 1995; McGarry et al., 2001; Zhou et al., 2003; Hush and Horne, 1993).

This type of network representation is difficult to understand due to the fact that a typical neural network contains a large number of characteristic value parameters. These parameters determine the relationship between the data input vectors and the output value *y*. Although the nature of parameter mapping for different training algorithms is understandable, a large number of typical network parameters make the task of understanding the nature of the network very difficult. Moreover, in multilayer networks,

such parameters characterise the various relationships between the data input vectors and the desired output values. In such cases, it is impossible to determine the effect of a given input vector on the output value because its effect is expressed in context with the values of other input vectors.

Figure 1 Rule extraction process from neural networks

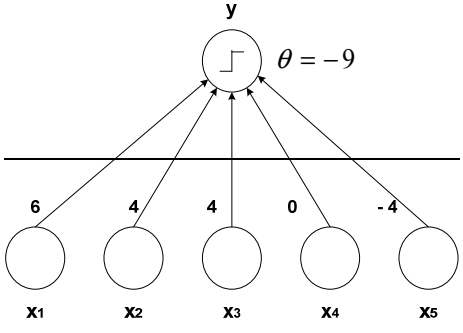


This type of network representation is difficult to understand due to the fact that a typical neural network contains a large number of characteristic value parameters.

Such relationships are assigned to the hidden layer elements of the network, which in a certain way combines many input vectors to obtain their characteristic values. It was hypothesised that exactly the elements of the hidden layer could characterise the relationship between the input data through the hidden layer to the output value of the network. What might this relationship be like? The hypothesis states that these could be expressions in a symbolic form that would describe such relationships, i.e., obtain the rules that characterise the network input and output relationships. Many researchers propose a variety of methods that address this hypothesis in the context of rule-making.

Figure 2 illustrates the task of obtaining rules from a simple network (Craven and Shavlik, 1995). It is a single layer network with 5 discrete input elements and one output element. The obtained symbolic rules characterise the state of the input elements, which when executed guarantee the correctness of the output state.

Figure 2 Simple network rule extraction example



For the input element, the activation value is assumed to be 0 if there is a false value and accordingly the activation value is 1 if there is a true value. The threshold function is used to calculate the output value:

$$y = \begin{cases} 1, \text{ if } \sum x_i w_i + \theta > 0 \\ 0, \quad \text{if else} \end{cases} \tag{1}$$

where y – output element activation, x_i – activation of input element i, w_i – weight value between input element i and output element, θ – output element threshold value.

In this case, the network of Figure 2 gives three rules that describe the general state of the network:

$$x_1 \wedge x_2 \wedge x_3 \rightarrow y$$
$$x_1 \wedge x_2 \wedge x_5 \rightarrow y$$
$$x_1 \wedge x_3 \wedge x_5 \rightarrow y.$$

Let's take rule 2 as an example – $x_1 \wedge x_2 \wedge x_5 \rightarrow y$. This rule states that in case $x_1 = true$, $x_2 = true$ and $x_5 = false$ and the output element y with an activation value of 1, the network predicts $y = true$. To make sure this is the correct rule, review the cases covered by this rule:

$$x_1 w_1 + x_2 w_2 + x_5 w_5 + \theta = 1 \tag{2}$$

The expression of weighted value is $sum > 0$. It is also seen that $0 \le x_3 w_3 + x_4 w_4 \le 4$. It does not matter what values the input elements x_3 and x_4 will accept – the output element will have an activation value of 1. So the rule found is correct and part of the rule condition is obtained. To make sure that the rule is as general as possible, we can remove one of the expressions in the rule conditional section. In this case, the rule will no longer accurately describe network behaviour. For example, if x_5 has been removed from the rule, the rule-compliant examples overlap the rule $-3 \le \sum x_i w_i + \theta \le 5$ and so the network does not allow output $y = true$ to be predicted in all cases.

This example illustrates the nature of rule enforcement in a very simple network. A question may arise – what does the rule mean for a network with a continuous activation function, hidden elements and lots of input elements? Although neural networks are mainly used to solve classification problems, they always have an implicit decision-making procedure that distinguishes each case to a particular class. In the example above, the decision procedure determines that $y = true$ if the output element activation is 1, and if $y = false$ the activation is 0. If the logistic activation function is used instead of the threshold function, then the decision procedure determines the case $y = true$, if the activation value exceeds a certain value.

Similarly, if a single output element is used to describe the separate class in multi-class assignments, the decision procedure associates the class with the output element that has the highest activation value. Generally, the resulting rule characterises the state of a set of network by a decision procedure and thus defines a particular class.

2.2 RBF and rule acquisition

The nature of the RBF neural network (see Figure 3) makes it an appropriate tool in the rule-making process. It is possible to get a number of IF-THEN rules that correctly describe the knowledge gained during the learning process.

The most frequently used radial function in networks is Gaussian:

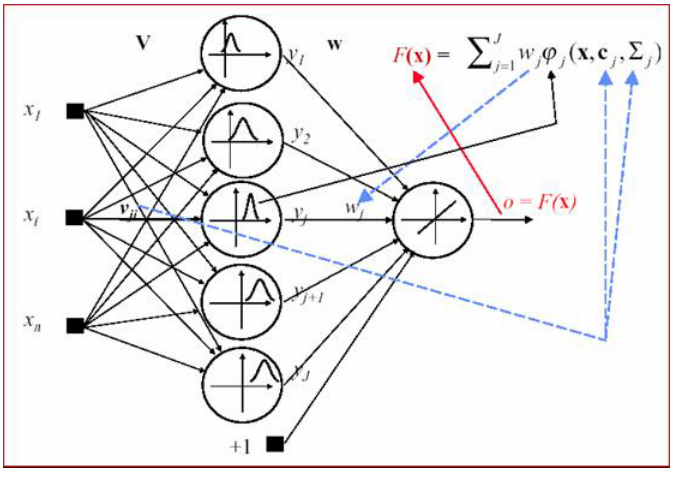
$$Z_j(x) = \exp\left(-\frac{\|x-\mu^2\|}{\delta_j^2}\right) \tag{3}$$

The response of the output unit:

$$y = \sum_{j=1}^{J} W_{ij} Z_j(x) \tag{4}$$

where W – weight matrix, Z – hidden units activations, x – input vectors, μ – parameter vector (centers), σ – width of receptive field.

Figure 3 Radial basic function network (see online version for colours)



The training of RBFs takes place in two stages – clustering (K-means) and supervised learning (LMS).

In the first step, the RBF centres are positioned using the K-means algorithm – hidden layer elements – and the radial function size.

As a result of algorithm operation, the final cluster centres w_j are determined, provided that the sum of the squares of the distances between all the points belonging to the group j and the cluster centre must be minimal.

In the second step of the learning, the weights from the hidden elements to the output and respectively the output response are calculated.

The rule-making process utilises the feature of hidden elements, namely, each hidden element, after training, actually represents one class of elements. The local nature of the RBF hidden element makes it possible to transform into a simple rule:

IF $Feature_1$ *is TRUE AND*

IF $Feature_2$ *is TRUE AND…*

IF $Feature_n$ *is TRUE*

THEN $Class_x$

where the complex element *Feature* consists of calculate the upper and lower limits of the RBF centres μ_n, RBF width or radius σ, steepness parameter S.

The value S is determined empirically and depends on the width parameter (Andrews and Gewa, 1995). The upper and lower limits are calculated using the formula:

$$X_{lower} = \mu_i - \sigma_i + S \quad \text{and} \quad X_{upper} = \mu_i + \sigma_i - S \tag{5}$$

3 The process of obtaining rules as a result of clustering

3.1 Illustrative example of rule acquisition using K-means clustering algorithm

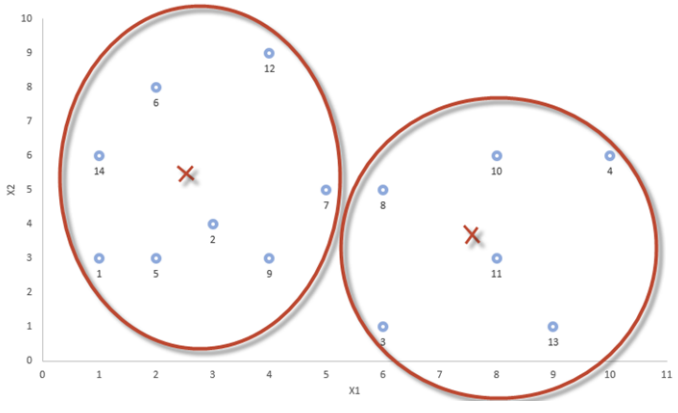
A two dimensional experimental dataset is given in Table 1.

Table 1 Experimental data

N	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
X_1	1	3	6	10	2	2	5	6	4	8	8	4	9	1
X_1	3	4	1	6	3	8	5	5	3	6	3	9	1	6

By using the RBF training algorithm, we derive two classes or clusters and their centres in three iterations after normalisation (see Figure 4). As a result, the following weight vectors were obtained: $\mu_1 = (2.75; 5.13)$ and $\mu_2 = (7.67; 3.67)$. There were also derived radius values $\sigma_1^2 = 6.8$ and $\sigma_2^2 = 7.4$ corresponding to the clusters. At the second stage of learning, radial functions and network output were calculated by formula (5). In this case RBF network is considered trained.

Figure 4 Two clusters with points (2.75; 5.13) and (7.67; 3.67) (see online version for colours)



The results obtained with SPSS program are also similar (see Table 2).

Table 2 SPSS results

Case number	*Cluster*	*Distance*
1	1	2.753
2	1	1.152
3	2	3.236
4	2	3.184
5	1	2.253
6	1	2.971
7	1	2.253
8	2	2.267
9	1	2.465
10	2	2.339
11	2	0.687
12	1	4.072
13	2	2.911
14	1	1.957

		Number of cases in each cluster
Cluster	1	8.000
	2	6.000
Valid		14.000
Missing		0

Steepness = 0.6:

Cluster 1. $X_{1\,lower} = 2.75 - 2.6 + 0.6 = 0.75$; $X_{2\,lower} = 5.13 - 2.6 + 0.6 = 3.13$;

$X_{1\,upper} = 2.75 + 2.6 - 0.6 = 4.95$; $X_{2\,upper} = 5.13 + 2.6 - 0.6 = 7.13$.

Cluster 2. $X_{1\,lower} = 7.67 - 2.7 + 0.6 = 5.57$; $X_{2\,lower} = 3.67 - 2.7 + 0.6 = 1.57$;

$X_{1\,upper} = 7.67 + 2.7 - 0.6 = 9.77$; $X_{2\,upper} = 3.67 + 2.7 - 0.6 = 5.77$.

We have derived the following rules:

$$IF\ (x_1 \geq 0.75\ AND \leq 4.95)\ AND\ IF\ (x_2 \geq 3.13\ AND \leq 7.13)\ THEN\ CLUSTER_1$$

$$IF\ (x_1 \geq 5.57\ AND \leq 9.77)\ AND\ IF\ (x_2 \geq 1.57\ AND \leq 5.77)\ THEN\ CLUSTER_2$$

Steepness = 0:

Cluster 1. $X_{1\,lower} = 2.75 - 2.6 + 0 = 0.15$; $X_{2\,lower} = 5.13 - 2.6 + 0 = 2.53$;

$X_{1\,upper} = 2.75 + 2.6 - 0 = 5.35$; $X_{2\,upper} = 5.13 + 2.6 - 0 = 7.73$.

Cluster 2. $X_{1\,lower} = 7.67 - 2.7 + 0 = 4.97$; $X_{2\,lower} = 3.67 - 2.7 + 0 = 0.97$;

$X_{1\,upper} = 7.67 + 2.7 - 0 = 10.37$; $X_{2\,upper} = 3.67 + 2.7 - 0 = 5.67$.

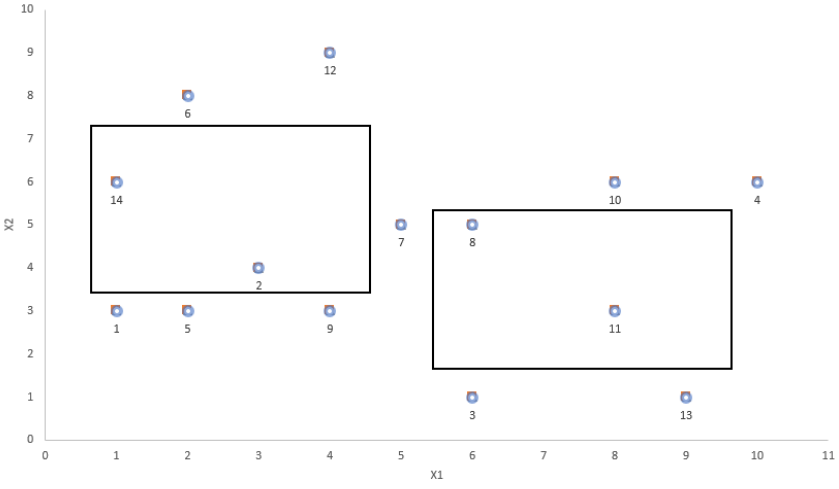
Thus, for each hidden unit that represents the clusters we have derived the following rules:

$$IF\ (x_1 \geq 0.15\ AND \leq 5.35)\ AND\ IF\ (x_2 \geq 2.53\ AND \leq 7.73)\ THEN\ CLUSTER_1$$

$$IF\ (x_1 \geq 4.97\ AND \leq 10.37)\ AND\ IF\ (x_2 \geq 0.97\ AND \leq 5.67)\ THEN\ CLUSTER_2$$

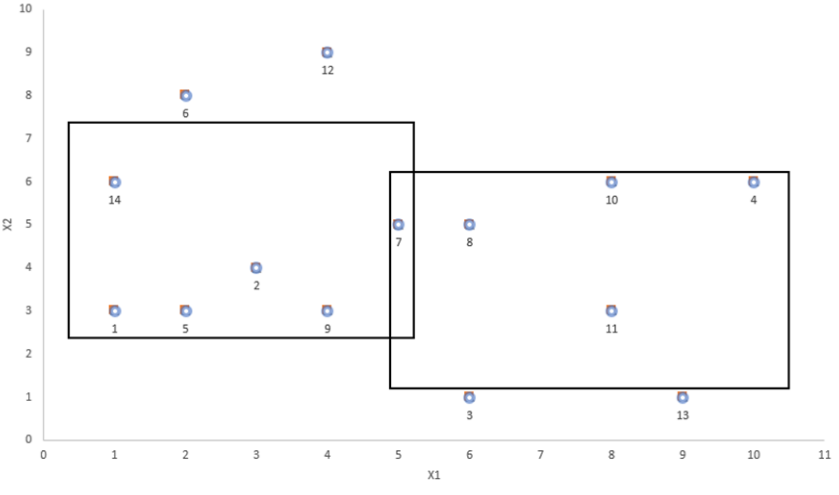
Regions of rules for *Steepness* value $S = 0.6$ are represented in Figure 5.

Figure 5 Regions of rules – *steepness* = 0.6 (see online version for colours)



Regions of rules for *Steepness* value $S = 0$ are represented in Figure 6.

Figure 6 Regions of rules – *steepness* = 0 (see online version for colours)



3.2 RULEX algorithm implementation with IRIS data

The experiment aimed at extracting the rules and testing their quality. During the experiment the well-known Fisher's IRIS dataset was employed (Iris Data set).

In the experimental part, all 50 elements of each cluster were taken as the training set. The values of parameter S were experimentally selected. In each training phase, cluster centers and radius values were calculated according to the RULEX algorithm. Based on these values, X_{lower} and X_{upper} and conditional parts of rules were calculated. Then a full IRIS data test was performed to see to what extent the rules found correctly describe the elements of each cluster. For each cluster, the number of validation elements as well as the percentage of the total number of correctly describing elements were found.

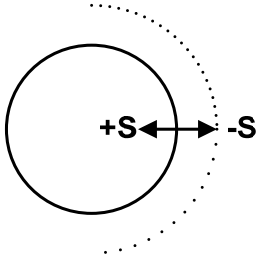
Table 3 demonstrates the results of the experiment.

Table 3 Results of training set

Parameter S	*Cluster I*	*Cluster II*	*Cluster III*	*Percentage* (%)
–0.8	49	50	49	98.7
–0.7	49	50	49	98.7
–0.6	48	49	49	97.3
–0.5	48	49	49	97.3
–0.4	45	47	47	92.7
–0.3	40	44	47	87.3
–0.2	40	42	44	84.0
–0.1	27	37	42	70.7
0	15	32	41	58.7
0.1	10	25	36	47.3
0.2	4	16	32	34.7
0.3	0	9	26	23.3

Analysing the obtained results, it can be concluded that parameter S play an important role – the higher the negative value of S, the more the lower limit X_{lower} of the rule's operating range is reduced while increasing the upper limit of the range X_{upper}. This effect is illustrated in Figure 7.

Figure 7 The effect of parameter *S* increase/decrease



4 Conclusions

In the experimental part implementation of the task with IRIS database was given. Using a rule-based design method at various initial parameters, data-specific rules were obtained.

As a result of the experiments, it was concluded that the resulting rules correctly describe the initial data and thus provide a basis for the evaluation of the methodology proposed in the paper. We were convinced that the different methods of extracting rules are able to find rules in the data and that the obtained rules are of good quality. Further the extracted rules can help discover and analyse the hidden knowledge in datasets. This will allow a decision to be made on the applicability of the resulting rules to different areas of decision making and analysis.

References

Andrews, R. and Gewa, S. (1995) 'RULEX and CEBP networks as the basic for a rule refinement system', *Hybrid Problems, Hybrid Solutions*, pp.1–12.

Andrews, R., Diederich, J. and Tickle, A. (1995) 'A survey and critique of techniques for extracting rules from trained artificial neural networks', *Knowledge-Based Systems*, Vol. 8, No. 6, pp.373–389, https://doi.org/10.1016/0950-7051(96)81920-4

Craven, M. and Shavlik, J. (1995) 'Extracting tree-structured representations of trained networks', *Advances in Neural Information Processing Systems*, Vol. 8.pp.24–30.

Duch, W., Setiono, R. and Zurada, J.M. (2004) 'Computational intelligence methods for rule-based data understanding', *Proceedings of IEEE*, Vol. 92, No. 5, pp.771–805, https://doi.org/10.1109/JPROC.2004.826605

Fausett, L. (1993) *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice Hall International Inc., New York.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996) 'From data mining to knowledge discovery databases', *AI Magazine*, Vol. 17, No. 3, pp.37–54, https://doi.org/10.1609/aimag.v17i3.1230

Hush, D.R. and Horne, B.G. (1993) 'Progress in supervised neural networks. what's new since Lippmann?', *IEEE Signal Processing Magazine*, Vol. 10, No 1.pp.32–43, https://doi.org/10.1109/79.180705

McGarry, K., Wermter, S. and Macintyre, J. (2001) 'The extraction and comparison of knowledge from local function networks', *International Journal of Computational Intelligence and Applications*, Vol. 1, No. 3, pp.369–382, https://doi.org/10.1142/S1469026801000305

Tan, P-N., Steinbach, M., Karpatne, A. and Kumar, V. (2019) *Introduction to Data Mining*, Pearson Education Limited, New York.

Zhou, Z.H., Jiang y. and Chen, S.F. (2003) 'Extracting symbolic rules from trained neural network ensembles', *AI Communications*, Vol. 16, No. 1, pp.3–15.

Website

Iris Data set – The UCI Machine Learning Repository [online] http://archive.ics.uci.edu/ml/datasets/Iris (Accessed 5 February, 2020).