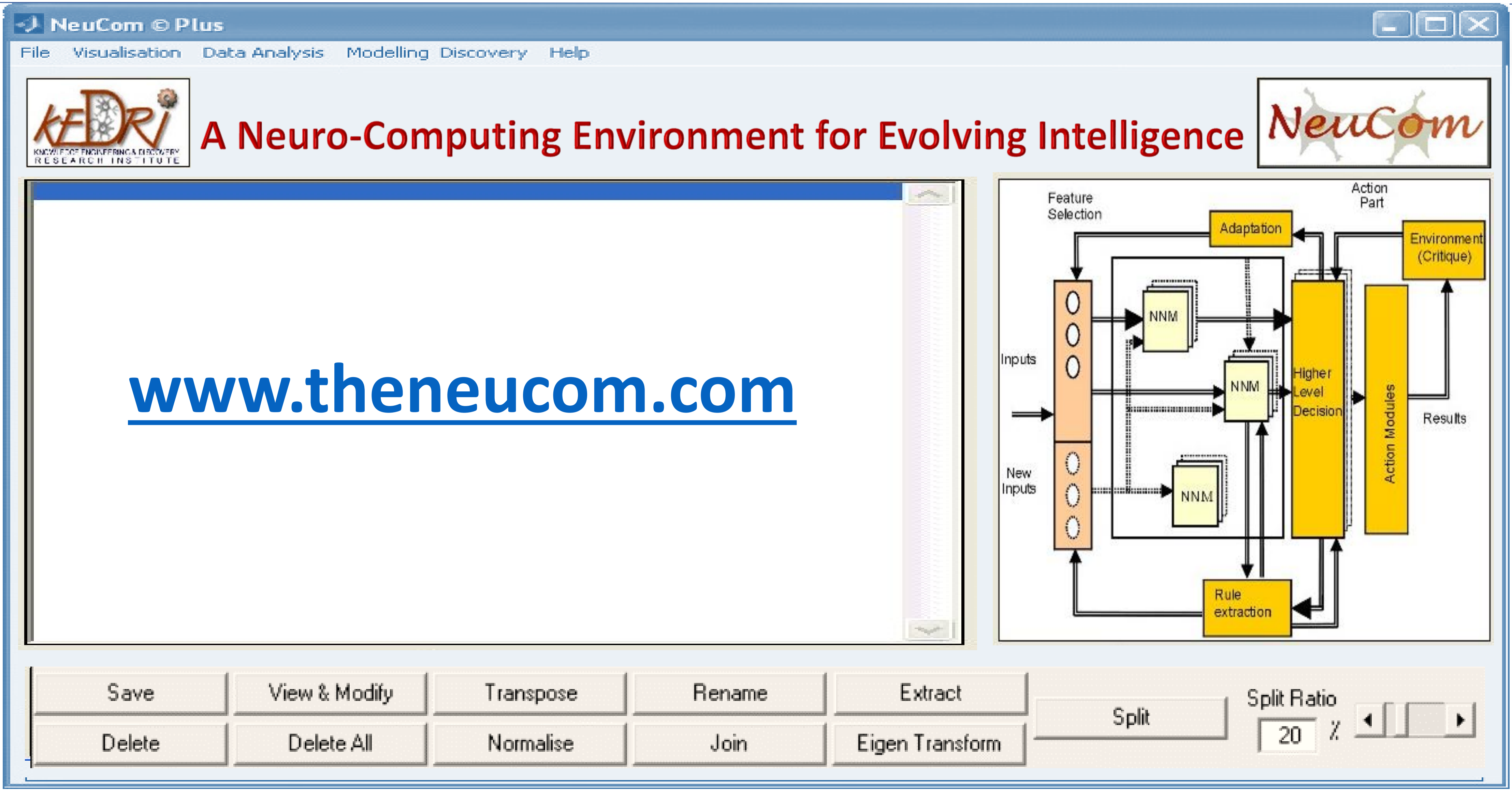
Evolving COnnectionist Systems (ECOS) - Principles

- ECOS are neural networks that learn from data and interact with their environment to evolve their structure, functionality, and internal knowledge representation.
- The learning process can be supervised, unsupervised, active, sleep/dream, forgetting/pruning, fuzzy rule insertion- and extraction-related, and so on.
- With little or no prior knowledge, "one-pass" training is used to quickly learn from large amounts of data.
- New input variables that are relevant to the task are allowed; and new outputs (classes), connections, and neurons are created/evolved.
- System self-evaluation in terms of behavior, global error and success, and knowledge representation.
- Applied to the development of various computational intelligence models, including evolving simple connectionist systems, evolving spiking neural networks, evolving rule-based and fuzzy systems, evolving kernel-based systems, evolving quantum-inspired systems, and many other integrated hybrid models.


NeuCom © Plus
File
Visualisation
Data Analysis
Modelling
Discovery
Help
A Neuro-Computing Environment for Evolving Intelligence
NeuCom
www.theneucom.com
Feature Selection
Action Part
Adaptation
Environment (Critique)
Inputs
New Inputs
NNM
NNM
NNM
Higher Level Decision
Action Modules
Results
Rule extraction
Save
View & Modify
Transpose
Rename
Extract
Split
Split Ratio
Delete
Delete All
Normalise
Join
Eigen Transform
20
%

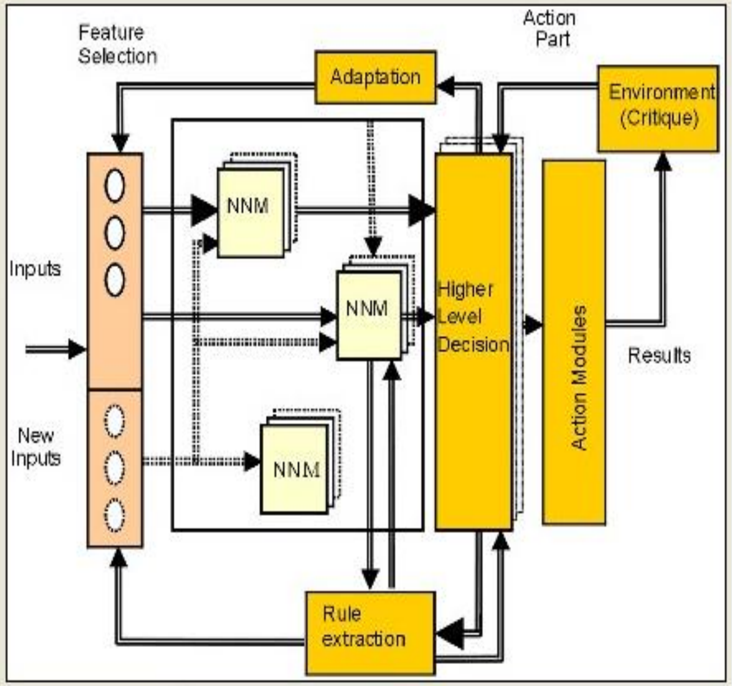
NeuCom © Plus

File Visualisation Data Analysis Modelling Discovery Help

www.theneucom.com

A Neuro-Computing Environment for Evolving Intelligence

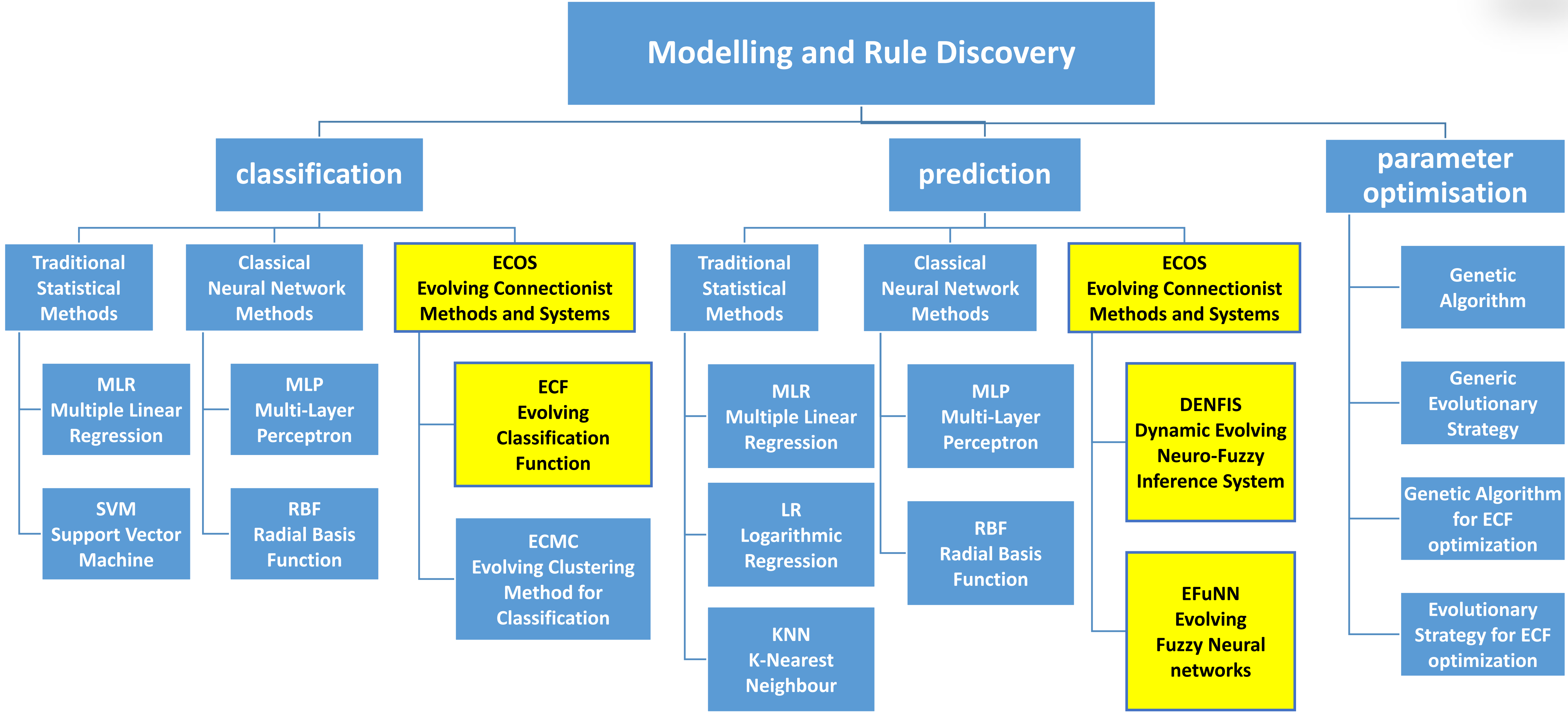
- NeuCom is a generic modular open architecture, based on ECOS principles, for fast data analysis, fast model prototyping, data development, patterns learning, knowledge discovery, and visualization.
- NeuCom algorithms use traditional and novel techniques for intelligent data analysis and system development.
- NeuCom provides incremental data learning, rule extraction, and data integration capabilities.
- NeuCom computational methods include feature selection, classification (Regression| MLP|ECF), prediction (Regression|MLP|EFuNN|DENFIS), and knowledge extraction.
- www.theneucom.com offers a free copy for research purposes.



Save | View & Modify | Transpose | Rename | Extract | Split | Split Ratio
Delete | Delete All | Normalise | Join | Eigen Transform | 20 %

NeuCom Modules

- data collection
- data pre-processing
  - normalization
  - filling missing values, …
- data visualization
  - 2D
  - 3D
  - PCA
  - surface plot, …
- Data analysis
  - correlation
  - signal to noise ratio,
  - Clustering: k-means, hierarchical-, evolving-.
- modelling and rule discovery
  - classification
  - prediction
  - optimisation
  - Cross Validation
- Documentation and online help.


Modelling and Rule Discovery
classification
prediction
parameter optimisation
Traditional Statistical Methods
Classical Neural Network Methods
ECOS Evolving Connectionist Methods and Systems
MLR Multiple Linear Regression
SVM Support Vector Machine
MLP Multi-Layer Perceptron
RBF Radial Basis Function
ECF Evolving Classification Function
ECMC Evolving Clustering Method for Classification
Traditional Statistical Methods
Classical Neural Network Methods
ECOS Evolving Connectionist Methods and Systems
MLR Multiple Linear Regression
LR Logarithmic Regression
KNN K-Nearest Neighbour
MLP Multi-Layer Perceptron
RBF Radial Basis Function
DENFIS Dynamic Evolving Neuro-Fuzzy Inference System
EFuNN Evolving Fuzzy Neural networks
Genetic Algorithm
Generic Evolutionary Strategy
Genetic Algorithm for ECF optimization
Evolutionary Strategy for ECF optimization

Evolving Classification Function

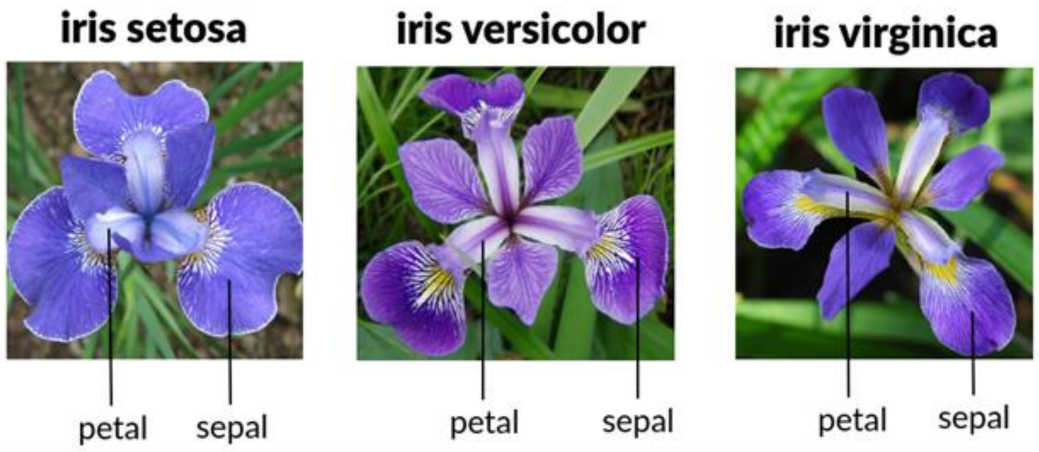
ECF Classification Model

IRIS Flower Classification Model
Evolving Classification Function [ECF]

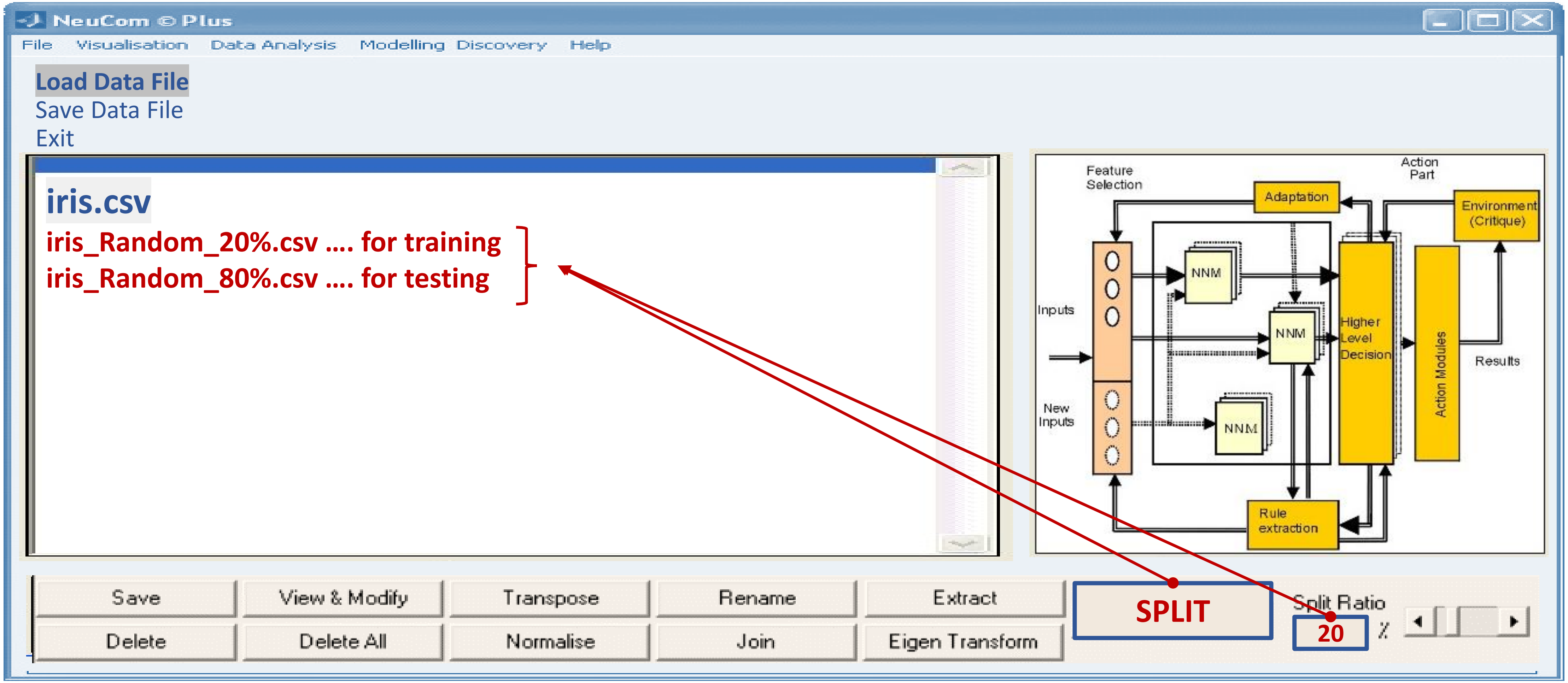
- The Iris flower dataset is widely used in machine learning and pattern recognition. It includes measurements of the sepal length/width and petal length/width of 150 samples of iris flowers, 50 samples from each of three different species: Iris setosa, Iris versicolor, and Iris virginica.
- Each sample in the dataset is labeled with the iris flower species, making this a supervised learning problem. The objective is to design an MLP model that can predict the species of an iris flower given its measurements.

ECF Classification Model: The Dataset

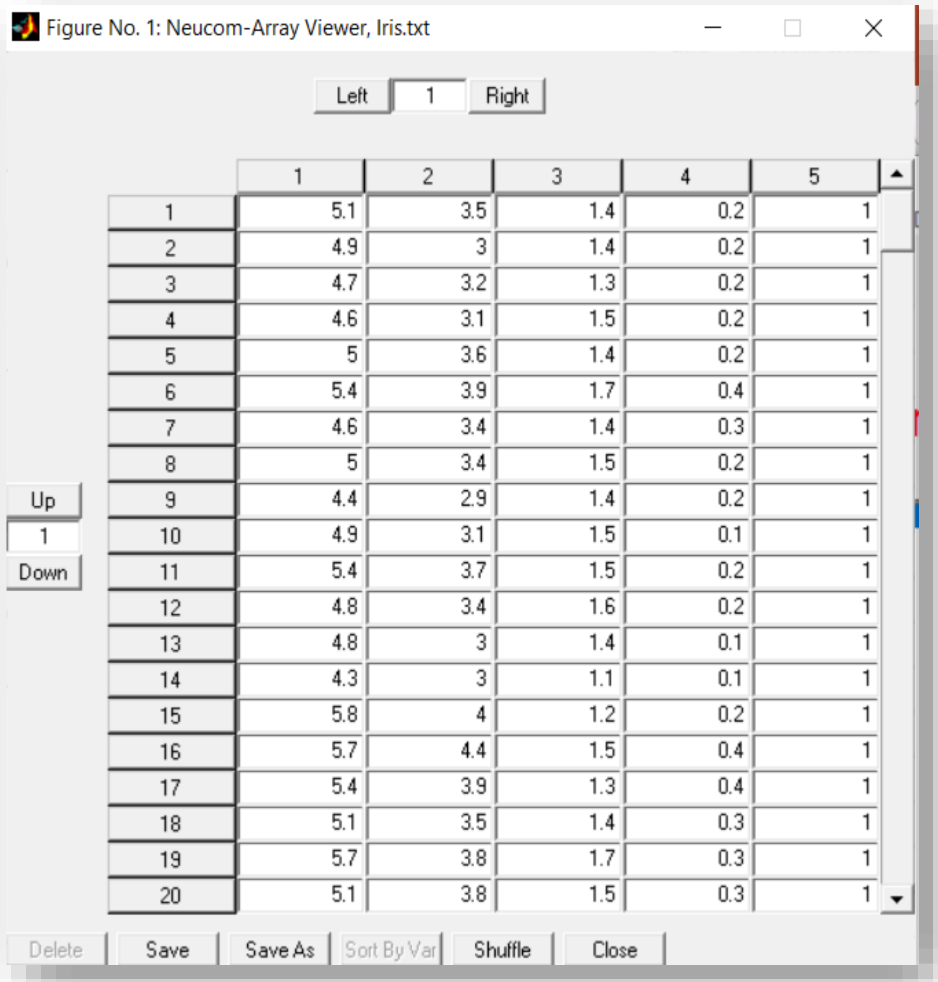
- Iris time series dataset consists of **150 iris plants.**
- **Four Features**: sepal length/width, petal length/width.
- **Three Classes**: Setosa, Versicolor, and Virginica.
- Iris dataset modelling for **classification.**
- Modelling method: **ECF**.
- Iris dataset **split** 20/80 training-learning ratio.



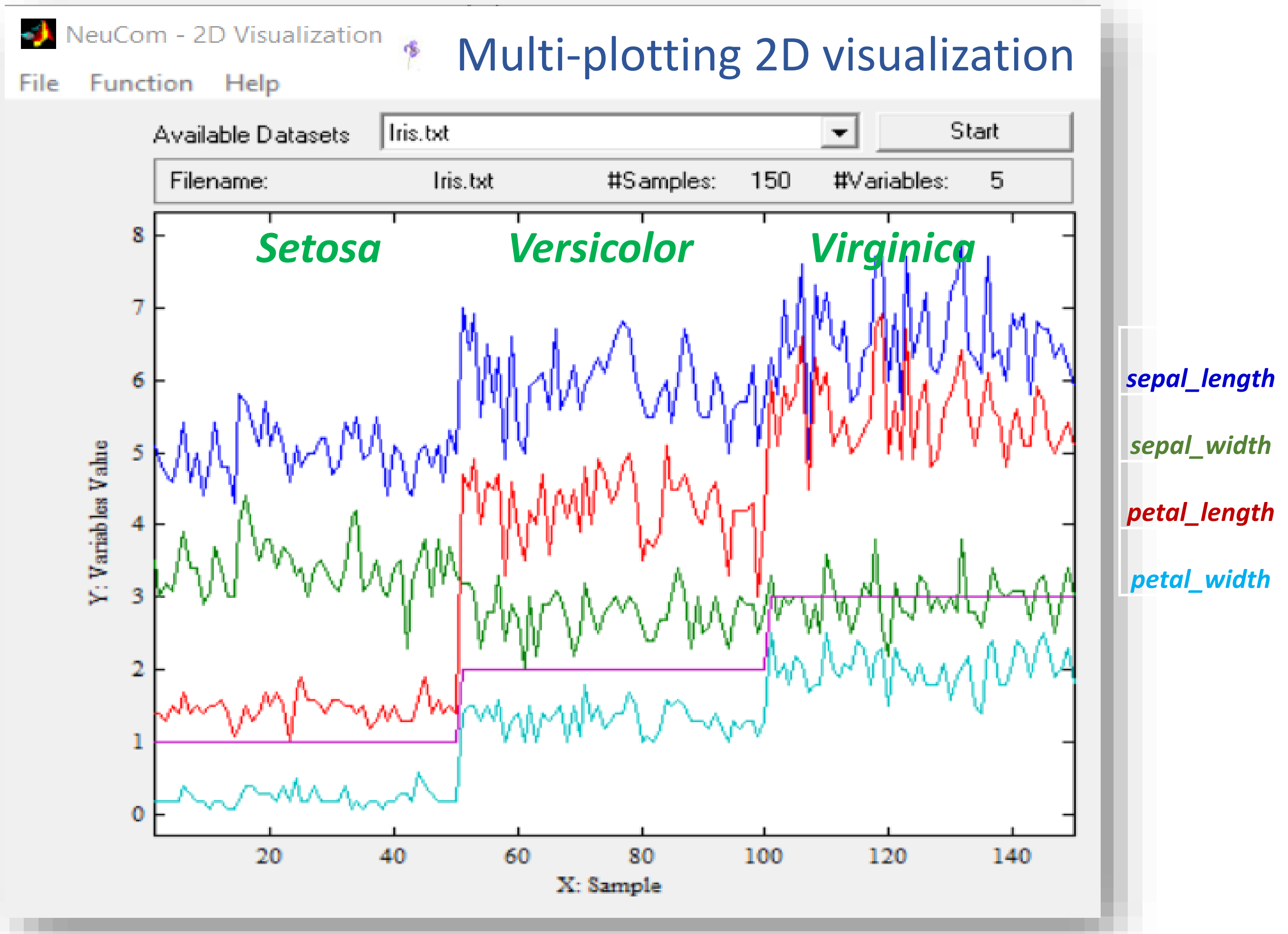
ECF Classification Model: Data Loading



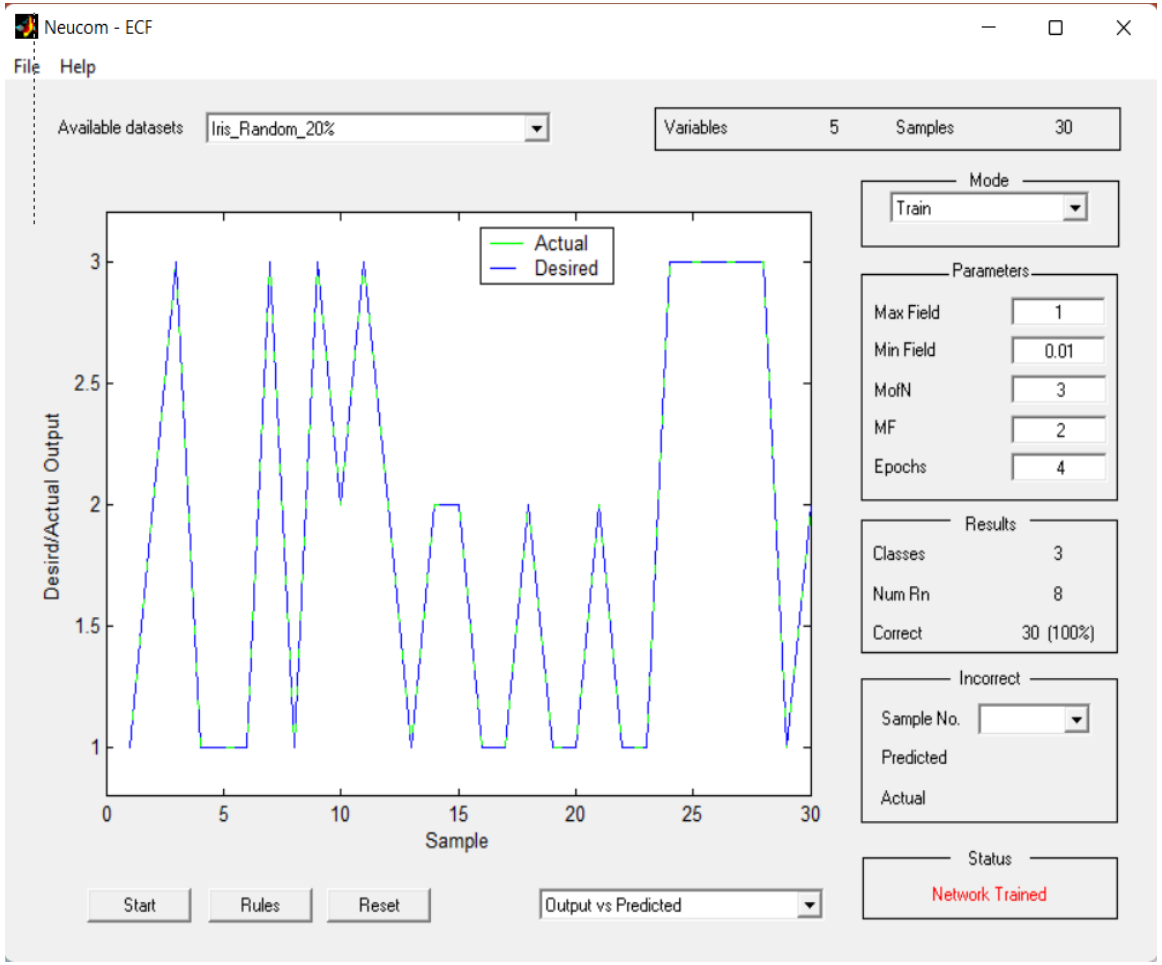
ECF Classification Model: Input and Visualisation



	1	2	3	4	5
1	5.1	3.5	1.4	0.2	1
2	4.9	3	1.4	0.2	1
3	4.7	3.2	1.3	0.2	1
4	4.6	3.1	1.5	0.2	1
5	5	3.6	1.4	0.2	1
6	5.4	3.9	1.7	0.4	1
7	4.6	3.4	1.4	0.3	1
8	5	3.4	1.5	0.2	1
9	4.4	2.9	1.4	0.2	1
10	4.9	3.1	1.5	0.1	1
11	5.4	3.7	1.5	0.2	1
12	4.8	3.4	1.6	0.2	1
13	4.8	3	1.4	0.1	1
14	4.3	3	1.1	0.1	1
15	5.8	4	1.2	0.2	1
16	5.7	4.4	1.5	0.4	1
17	5.4	3.9	1.3	0.4	1
18	5.1	3.5	1.4	0.3	1
19	5.7	3.8	1.7	0.3	1
20	5.1	3.8	1.5	0.3	1



ECF Classification Model: Parameters, Modelling and Analysis



Parameters:

- **MaxField** If the distance between a new sample (to be used for training) and every cluster centre is greater than this value, a new RN is created.
- **MinField** Initial radius of a newly created RN (rule nodes).
- **MofN** Number of nodes which are referenced to determine the class of the current sample.
- **MF** Number of Membership Functions (MFs) which are used to fuzzify the input data.
- **Epochs** Number of learning iterations used by the network.

Results:

- **Classes** Number of classes which the network distinguishes between.
- **Num Rn** Number of rule nodes in the networks structure.
- **Correct** Number of samples which have been successfully classified by the network.
- **Incorrect** Listing only samples which have been incorrectly classified.

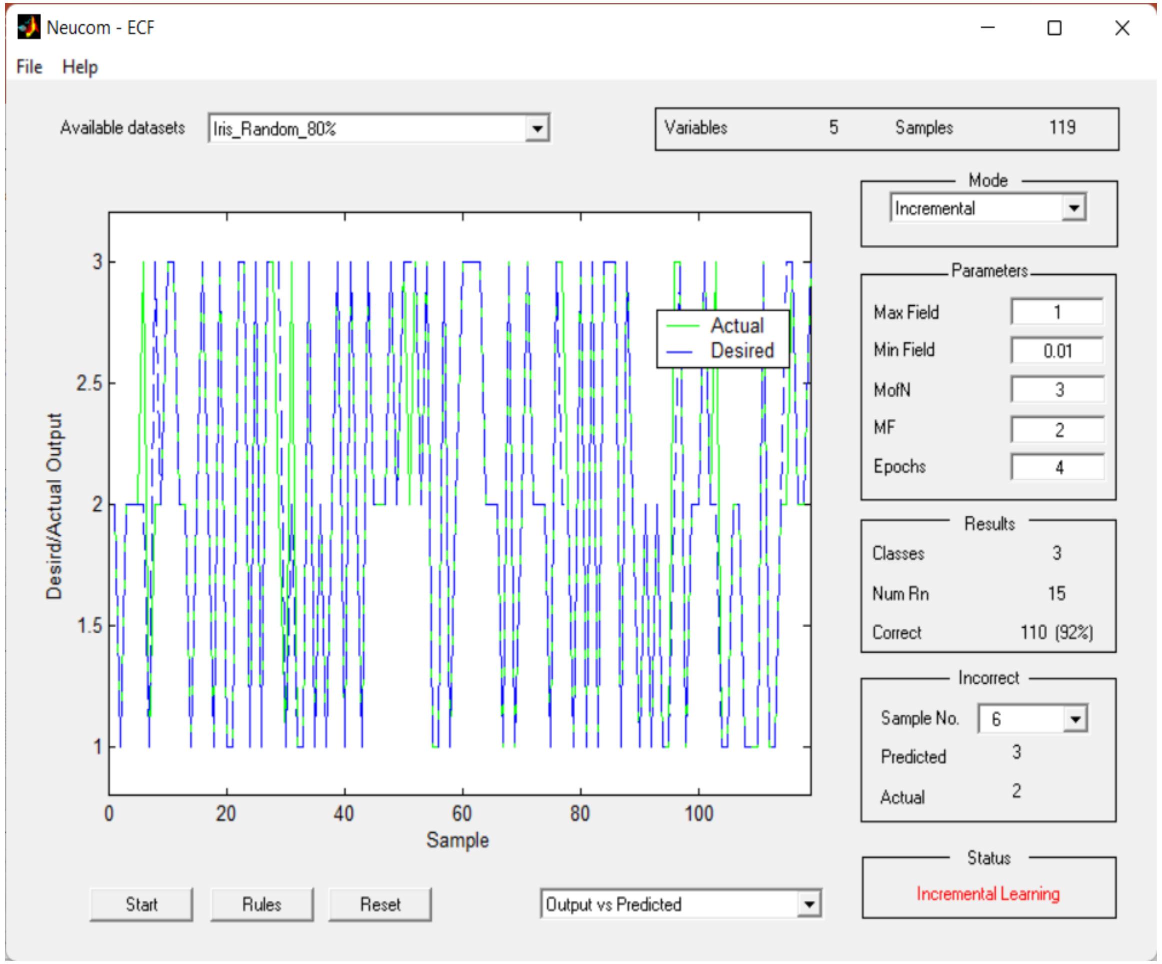
Options:

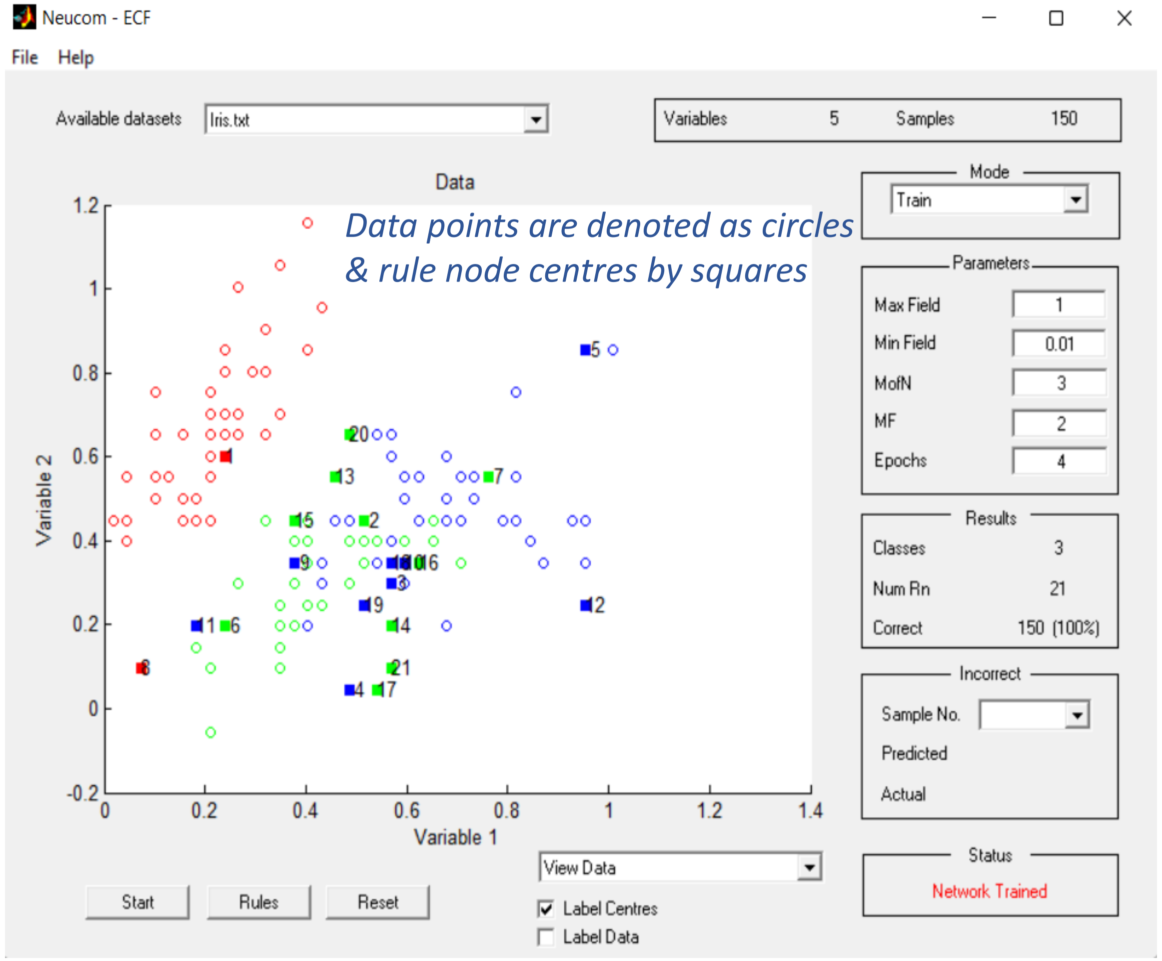
- **Start**: Begin which ever is currently displayed in the mode menu (training, testing, etc.)
- **Rules**: Extract and display the rules which have been created by the network
- **Reset**: Delete the current network from memory.

Graphical presentation:

- **Output vs predicted**: plots network output vs desired output as dictated by the data set.
- **%Accuracy/class:** Displays in a bar chart the percentage of each class which have been correctly classified.
- **No of rule nodes:** Number of rule nodes dedicated to each class by the model.
- **View data:** Graphical representation of the data (circles) and the rule node centres (squares).
- **RN radius:** Displays radii of each rule node. The rule nodes are grouped together according to their class.
- **Confusion table:** Displays in compact format the correct and incorrect classifications made by the system.

ECF Classification Model: Parameters, Modelling and Analysis





ECF Classification Model: Rules extraction

Rule 1:
if
X1 is (1: 0.76)
X2 is (2: 0.60)
X3 is (1: 0.87)
X4 is (1: 0.84)
then Class is [1]
Radius = 0.253164 , 208 in Cluster

Rule 2:
if
X1 is (1: 0.93)
X2 is (1: 0.90)
X3 is (1: 0.94)
X4 is (1: 0.92)
then Class is [1]
Radius = 0.010000 , 5 in Cluster

Rule 3:
if
X1 is (2: 0.51)
X2 is (1: 0.55)
X3 is (2: 0.61)
X4 is (2: 0.52)
then Class is [2]
Radius = 0.072538 , 56 in Cluster

Rule 4:
if
X1 is (1: 0.76)
X2 is (1: 0.80)
X3 is (1: 0.66)
X4 is (1: 0.60)
then Class is [2]
Radius = 0.172745 , 54 in Cluster

Rule 5:
if
X1 is (2: 0.76)
X2 is (2: 0.55)
X3 is (2: 0.62)
X4 is (2: 0.52)
then Class is [2]
Radius = 0.107222 , 24 in Cluster

Rule 6:
if
X1 is (1: 0.54)
X2 is (2: 0.55)
X3 is (2: 0.64)
X4 is (2: 0.68)
then Class is [2]
Radius = 0.010000 , 4 in Cluster

Rule 7:
if
X1 is (2: 0.57)
X2 is (1: 0.80)
X3 is (2: 0.66)
X4 is (2: 0.56)
then Class is [2]
Radius = 0.070157 , 4 in Cluster

Rule 8:
if
X1 is (1: 0.62)
X2 is (1: 0.55)
X3 is (2: 0.52)
X4 is (1: 0.52)
then Class is [2]
Radius = 0.126069 , 52 in Cluster

Rule 9:
if
X1 is (2: 0.62)
X2 is (1: 0.65)
X3 is (2: 0.61)
X4 is (2: 0.56)
then Class is [2]
Radius = 0.010000 , 4 in Cluster

Rule 10:
if
X1 is (2: 0.54)
X2 is (1: 0.95)
X3 is (2: 0.59)
X4 is (2: 0.56)
then Class is [2]
Radius = 0.048663 , 3 in Cluster

Rule 11:
if
X1 is (1: 0.51)
X2 is (2: 0.65)
X3 is (2: 0.59)
X4 is (2: 0.60)
then Class is [2]
Radius = 0.010000 , 3 in Cluster

Rule 12:
if
X1 is (2: 0.57)
X2 is (1: 0.90)
X3 is (2: 0.57)
X4 is (1: 0.52)
then Class is [2]
Radius = 0.010000 , 4 in Cluster

Rule 13:
if
X1 is (2: 0.57)
X2 is (1: 0.70)
X3 is (2: 0.66)
X4 is (2: 0.68)
then Class is [3]
Radius = 0.040904 , 12 in Cluster

Rule 14:
if
X1 is (1: 0.51)
X2 is (1: 0.95)
X3 is (2: 0.67)
X4 is (2: 0.56)
then Class is [3]
Radius = 0.010000 , 5 in Cluster

Rule 15:
if
X1 is (2: 0.95)
X2 is (2: 0.85)
X3 is (2: 0.95)
X4 is (2: 0.84)
then Class is [3]
Radius = 0.281771 , 63 in Cluster

Rule 16:
if
X1 is (1: 0.62)
X2 is (1: 0.65)
X3 is (2: 0.66)
X4 is (2: 0.76)
then Class is [3]
Radius = 0.099973 , 35 in Cluster

Rule 17:
if
X1 is (2: 0.60)
X2 is (1: 0.65)
X3 is (2: 0.77)
X4 is (2: 0.80)
then Class is [3]
Radius = 0.105996 , 47 in Cluster

Rule 18:
if
X1 is (1: 0.82)
X2 is (1: 0.80)
X3 is (2: 0.59)
X4 is (2: 0.64)
then Class is [3]
Radius = 0.010000 , 4 in Cluster

Rule 19:
if
X1 is (2: 0.95)
X2 is (1: 0.75)
X3 is (2: 0.99)
X4 is (2: 0.88)
then Class is [3]
Radius = 0.259563 , 37 in Cluster

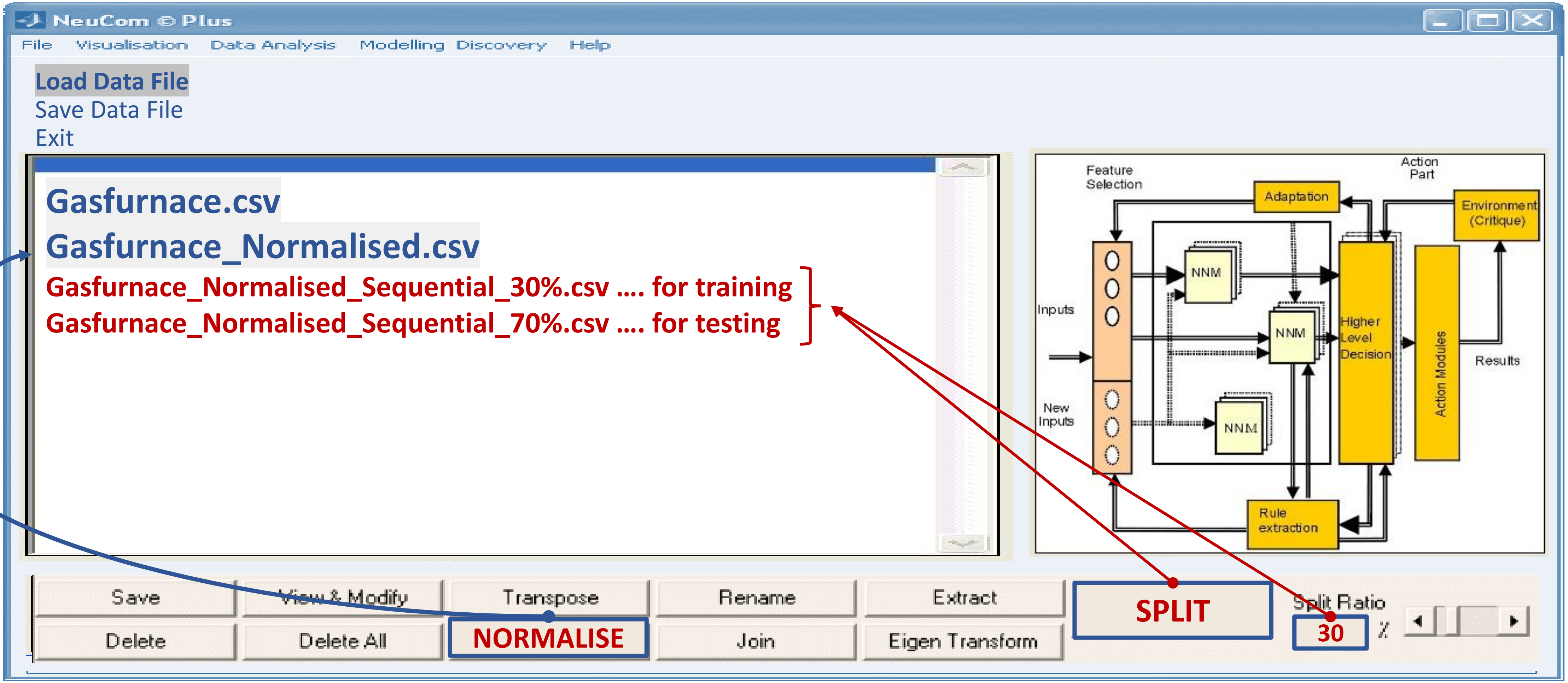
Rule 20:
if
X1 is (2: 0.57)
X2 is (1: 0.65)
X3 is (2: 0.69)
X4 is (2: 0.56)
then Class is [3]
Radius = 0.010000 , 3 in Cluster

Rule 21:
if
X1 is (2: 0.51)
X2 is (1: 0.75)
X3 is (2: 0.77)
X4 is (2: 0.52)
then Class is [3]
Radius = 0.010000 , 3 in Cluster

GasFurnace Prediction Model

- The gas furnace dataset for time series analysis contains the gas rate and the percentage CO_2 in the gas.
- Gas furnace time series dataset consists of **292 observations.**
- **Two input features:** Methane and CO_2
- **One output feature:** $CO_2(t+1) = f\,(\text{Methane}(t-4), CO_2(t)\ , \varepsilon)$
- dataset modelling for **Prediction.**
- Modelling method: **MLP**.
- Gas furnace dataset **split** 30/70 training-learning ratio.

GasFurnace Prediction Model: Data Loading



GasFurnace Prediction Model: Input and Visualization

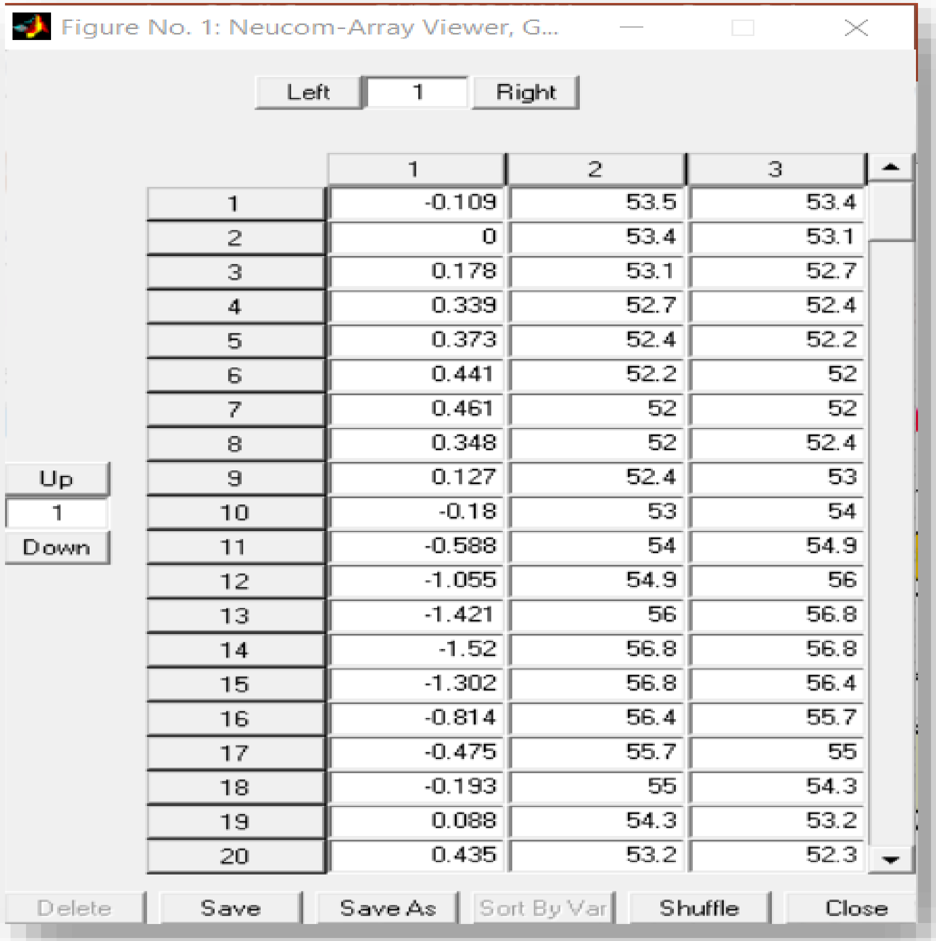


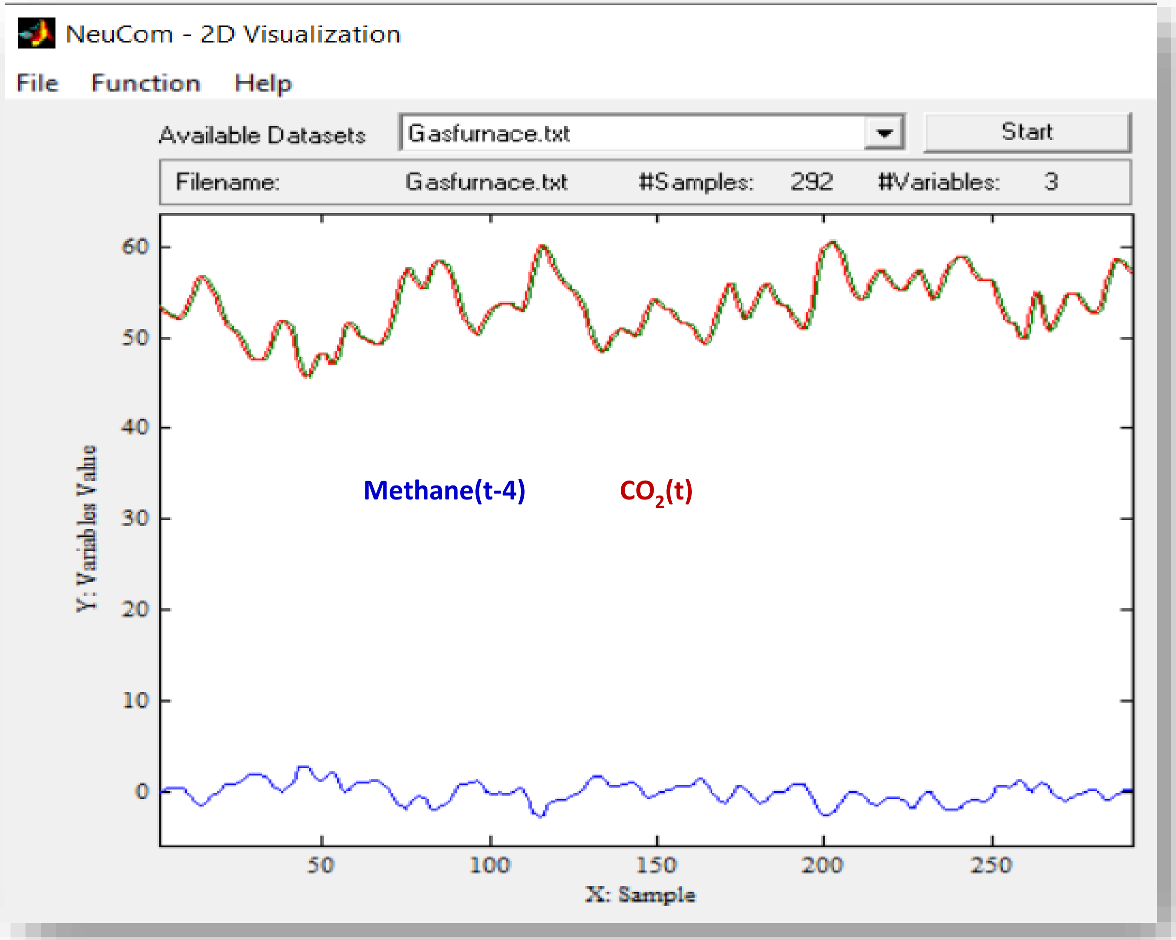
Figure No. 1: Neucom-Array Viewer, G...

Left 1 Right

	1	2	3
1	-0.109	53.5	53.4
2	0	53.4	53.1
3	0.178	53.1	52.7
4	0.339	52.7	52.4
5	0.373	52.4	52.2
6	0.441	52.2	52
7	0.461	52	52
8	0.348	52	52.4
9	0.127	52.4	53
10	-0.18	53	54
11	-0.588	54	54.9
12	-1.055	54.9	56
13	-1.421	56	56.8
14	-1.52	56.8	56.8
15	-1.302	56.8	56.4
16	-0.814	56.4	55.7
17	-0.475	55.7	55
18	-0.193	55	54.3
19	0.088	54.3	53.2
20	0.435	53.2	52.3

Up 1 Down

Delete Save Save As Sort By Var Shuffle Close



Dynamic Evolving Neuro-Fuzzy Inference System

DENFIS Prediction Model

Dynamic Evolving Neuro-Fuzzy Inference System
DENFIS Prediction Model: Parameters, Modelling and Analysis

Parameters:

- **Dthr**: Distance Threshold which determines the maximum radius of the rule nodes in this network.
- **M-of-N**: Number of nodes which are referenced to estimate the output of the current sample.
- **Epochs**: Number of learning iterations used to train the network

Results:

- **NumRn**: Number of Rule Nodes (RNs) in the network.
- **NDEI**: Non-Dimensional Error Index, defined as RMSE / StDev of the target series.
- **RMSE**: Root Mean Squared Error.
- **Incorrect** Listing samples which have been incorrectly predicted.

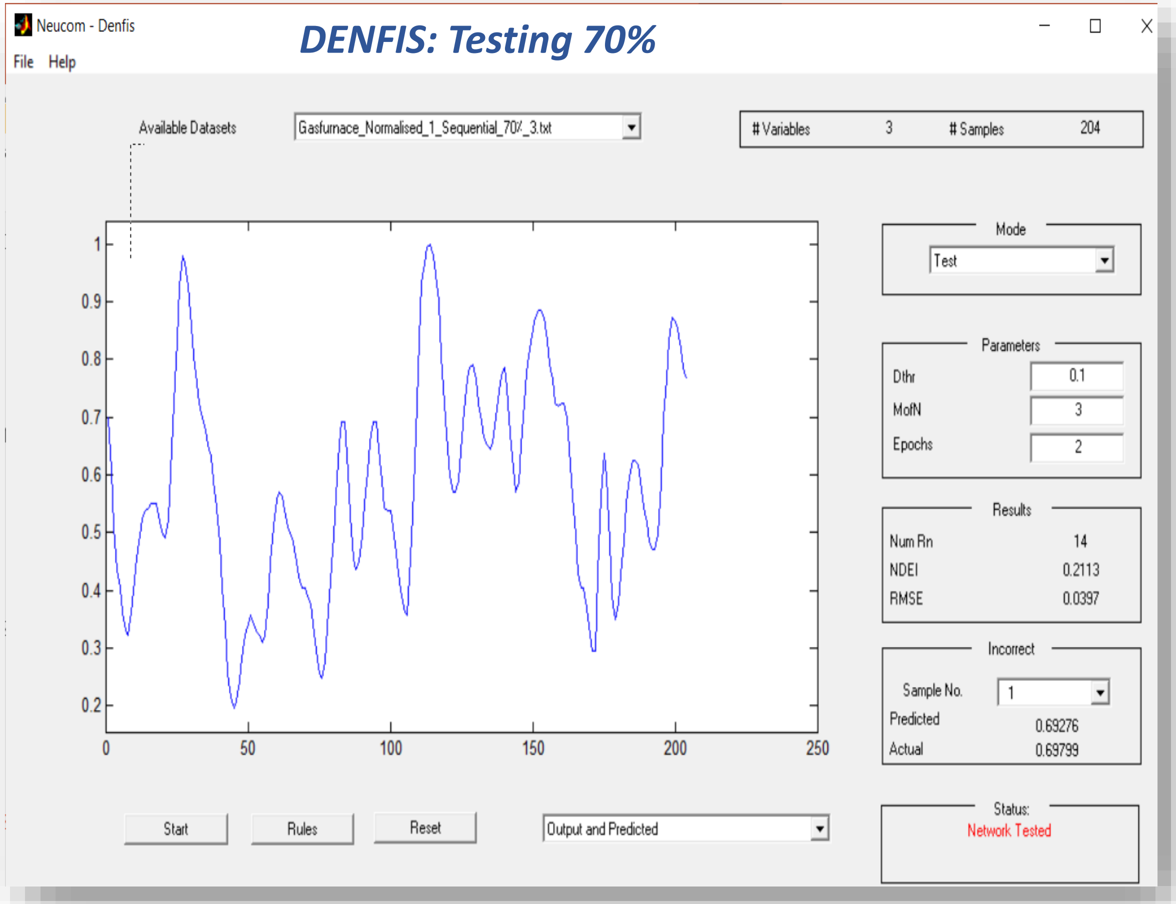
Options:

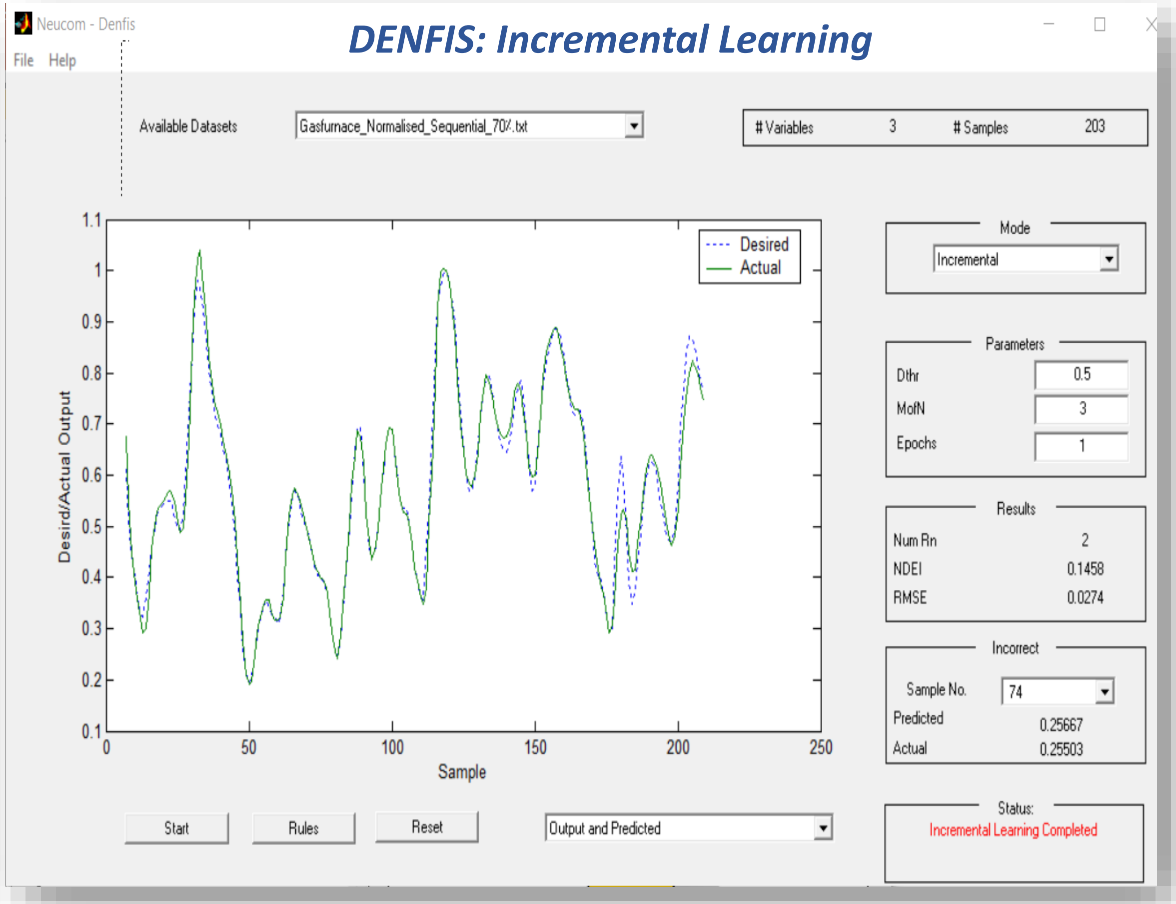
- **Start**: Begin whichever is currently displayed in the mode menu (training, testing, etc.)
- **Rules**: Extract and display the rules which have been created by the network
- **Reset**: Delete the current network from memory.

Graphical presentation:

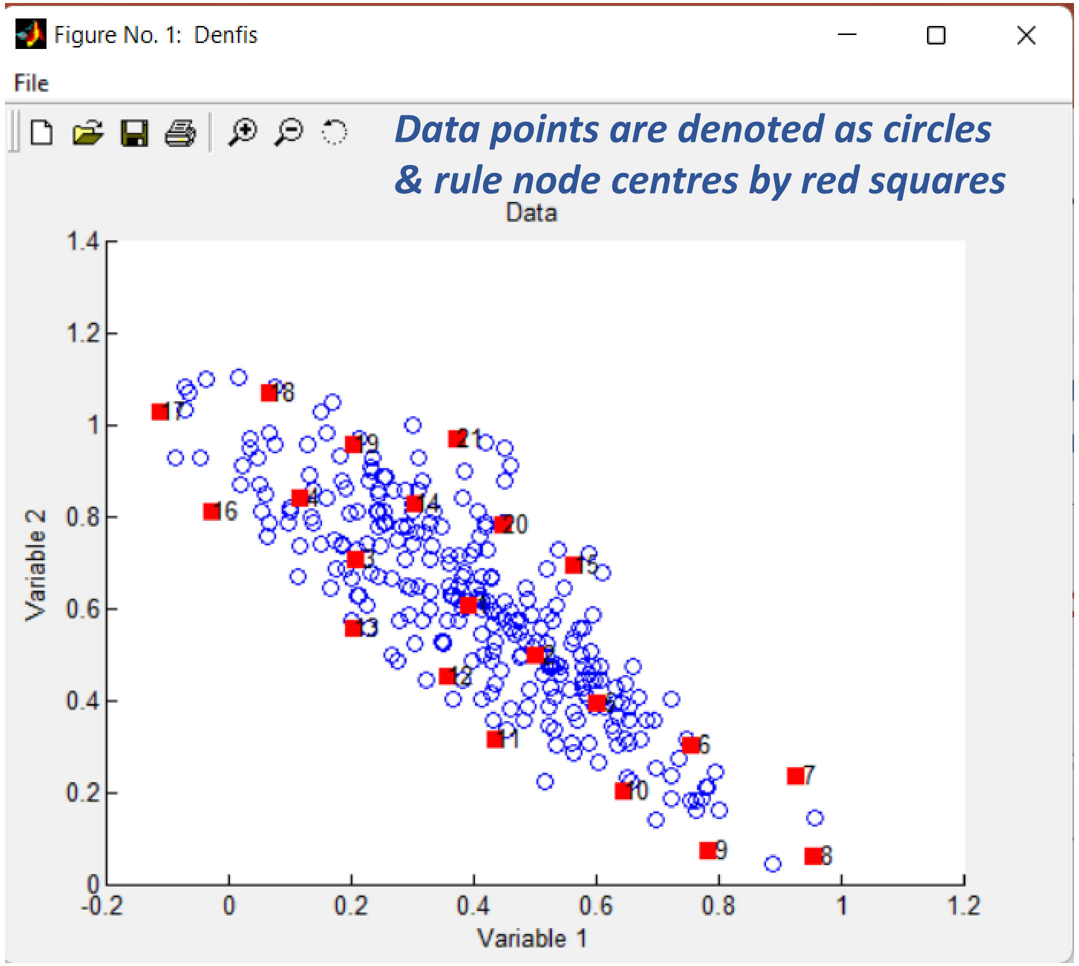
- **Output and Predicted**: The green line indicates the output produced by the network and the dotted line indicates the desired output.
- **ABSE Error**: This shows the Absolute Error for each sample.

DENFIS Prediction Model: Parameters, Modelling and Analysis





DENFIS Prediction Model: Rules extraction



Rule 1:
if X1 is GaussianMF(0.50 0.39)
X2 is GaussianMF(0.50 0.61)
then Y = 1.54 - 0.56 * X1 + 0.49 * X2

Rule 2:
if X1 is GaussianMF(0.50 0.50)
X2 is GaussianMF(0.50 0.50)
then Y = 1.53 - 0.60 * X1 + 0.52 * X2

Rule 3:
if X1 is GaussianMF(0.50 0.21)
X2 is GaussianMF(0.50 0.71)
then Y = 1.44 - 0.40 * X1 + 0.57 * X2

Rule 4:
if X1 is GaussianMF(0.50 0.12)
X2 is GaussianMF(0.50 0.84)
then Y = 1.48 - 0.45 * X1 + 0.52 * X2

Rule 5:
if X1 is GaussianMF(0.50 0.60)
X2 is GaussianMF(0.50 0.39)
then Y = 1.47 - 0.53 * X1 + 0.59 * X2

Rule 6:
if X1 is GaussianMF(0.50 0.75)
X2 is GaussianMF(0.50 0.30)
then Y = 1.46 - 0.47 * X1 + 0.49 * X2

Rule 7:
if X1 is GaussianMF(0.50 0.92)
X2 is GaussianMF(0.50 0.24)
then Y = 1.51 - 0.50 * X1 + 0.33 * X2

Rule 8:
if X1 is GaussianMF(0.50 0.95)
X2 is GaussianMF(0.50 0.06)
then Y = 1.51 - 0.51 * X1 + 0.34 * X2

Rule 9:
if X1 is GaussianMF(0.50 0.78)
X2 is GaussianMF(0.50 0.07)
then Y = 1.49 - 0.49 * X1 + 0.42 * X2

Rule 10:
if X1 is GaussianMF(0.50 0.64)
X2 is GaussianMF(0.50 0.20)
then Y = 1.52 - 0.52 * X1 + 0.36 * X2

Rule 11:
if X1 is GaussianMF(0.50 0.43)
X2 is GaussianMF(0.50 0.32)
then Y = 1.49 - 0.56 * X1 + 0.54 * X2

Rule 12:
if X1 is GaussianMF(0.50 0.36)
X2 is GaussianMF(0.50 0.45)
then Y = 1.39 - 0.39 * X1 + 0.63 * X2

Rule 13:
if X1 is GaussianMF(0.50 0.20)
X2 is GaussianMF(0.50 0.56)
then Y = 1.39 - 0.46 * X1 + 0.65 * X2

Rule 14:
if X1 is GaussianMF(0.50 0.30)
X2 is GaussianMF(0.50 0.83)
then Y = 1.18 - 0.34 * X1 + 0.88 * X2

Rule 15:
if X1 is GaussianMF(0.50 0.56)
X2 is GaussianMF(0.50 0.69)
then Y = 1.34 - 0.42 * X1 + 0.72 * X2

Rule 16:
if X1 is GaussianMF(0.50 -0.03)
X2 is GaussianMF(0.50 0.81)
then Y = 1.53 - 0.51 * X1 + 0.47 * X2

Rule 17:
if X1 is GaussianMF(0.50 -0.11)
X2 is GaussianMF(0.50 1.03)
then Y = 1.50 - 0.51 * X1 + 0.51 * X2

Rule 18:
if X1 is GaussianMF(0.50 0.06)
X2 is GaussianMF(0.50 1.07)
then Y = 1.42 - 0.44 * X1 + 0.60 * X2

Rule 19:
if X1 is GaussianMF(0.50 0.20)
X2 is GaussianMF(0.50 0.95)
then Y = 1.32 - 0.28 * X1 + 0.69 * X2

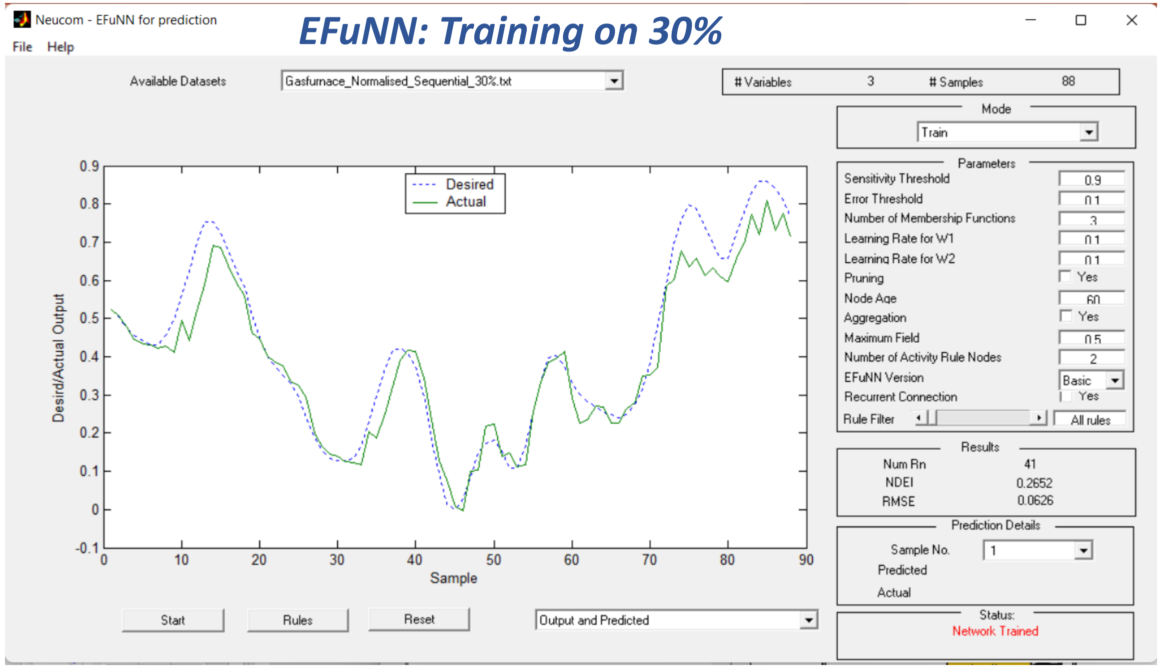
Rule 20:
if X1 is GaussianMF(0.50 0.45)
X2 is GaussianMF(0.49 0.78)
then Y = 1.31 - 0.40 * X1 + 0.75 * X2

Rule 21:
if X1 is GaussianMF(0.50 0.37)
X2 is GaussianMF(0.50 0.97)
then Y = 1.05 - 0.22 * X1 + 1.00 * X2

Evolving Fuzzy Neural Networks

EFuNN Prediction Model

Evolving Fuzzy Neural Networks
EFuNN Prediction Model: Parameters, Modelling and Analysis



Graphical presentation:

- **Output and Predicted**: The green line indicates the output produced by the network and the dotted line indicates the desired output.
- **ABSE Error**: This shows the Absolute Error for each sample.
- **Show Data**: Plots the data and rule nodes where appropriate.

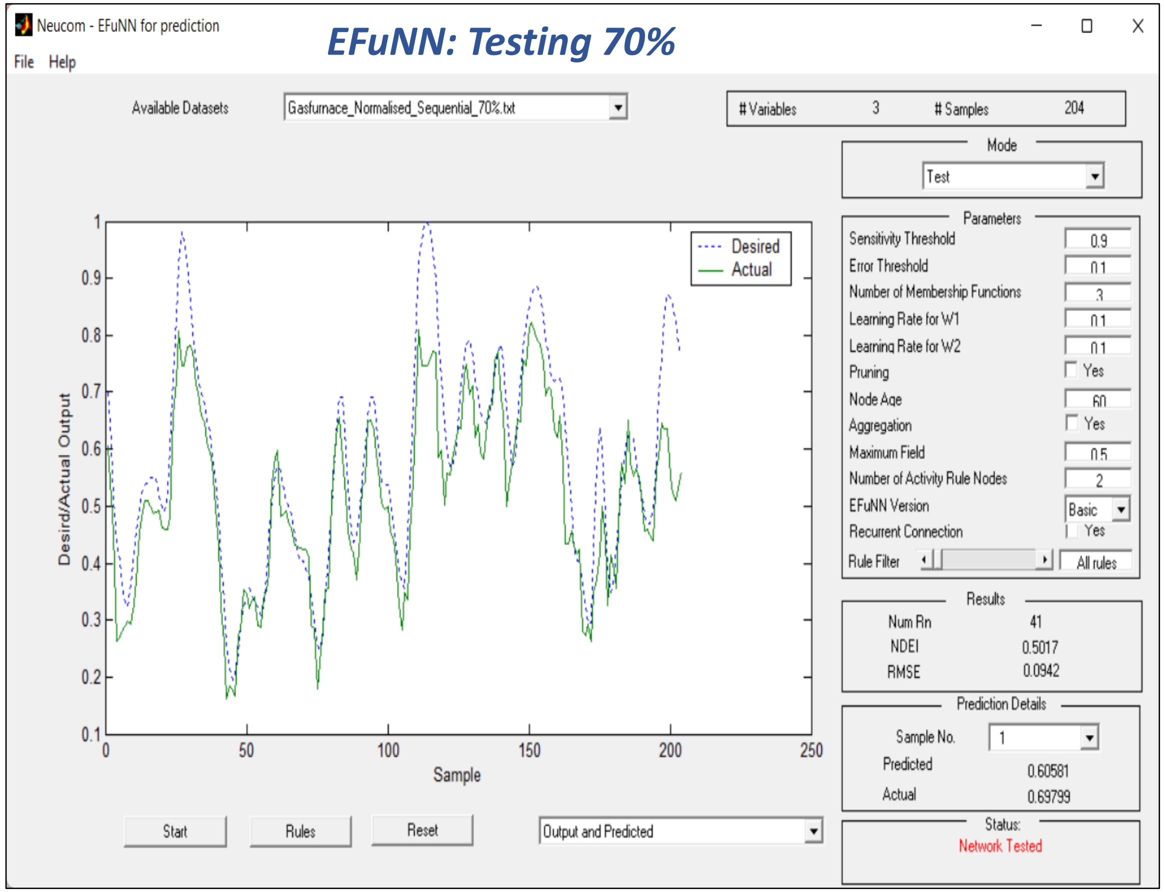
Parameters:

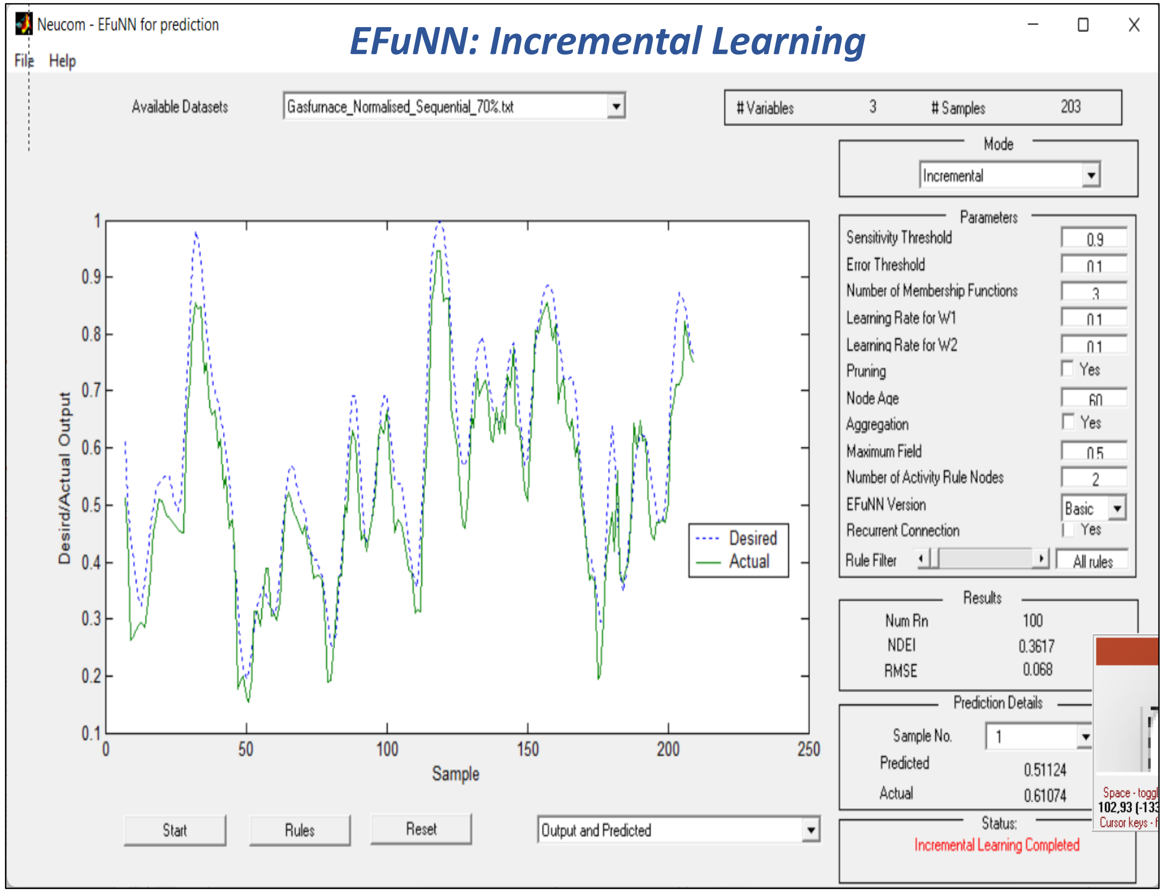
- **Sensitivity Threshold**: Maximum cluster radius (rule nodes)
- **Error Threshold**: Level of error tolerance for the output
- **Number of Membership Function**: No. of membership function used in fuzzy inference system.
- **Learning Rate for W1**: Learning Rate for the weights of first layer
- **Learning Rate for W2**: Learning Rate for the weights of second layer
- **Pruning**: Whether to prune old nodes
- **Node Age**: If Pruning is enabled, then old nodes should be removed
- **Aggregation**: Whether to allow similar rule nodes to be merged.
- **Max Field**: Maximum radius for clusters in self-tuning mode. (only applied in self-tuning mode)
- **Number of Activity Rule Nodes**: Number of rule nodes used to derive the output
- **EFuNN Version**: Standard or Self-tuning EFuNN
- **Recurrent Connection**: Whether to allow recurrent connections

Results:

- **NumRn**: Number of Rule Nodes (RNs) in the network.
- **NDEI**: Non-Dimensional Error Index, defined as RMSE / StDev of the target series.
- **RMSE**: Root Mean Squared Error.
- **Incorrect** Listing samples which have been incorrectly predicted.

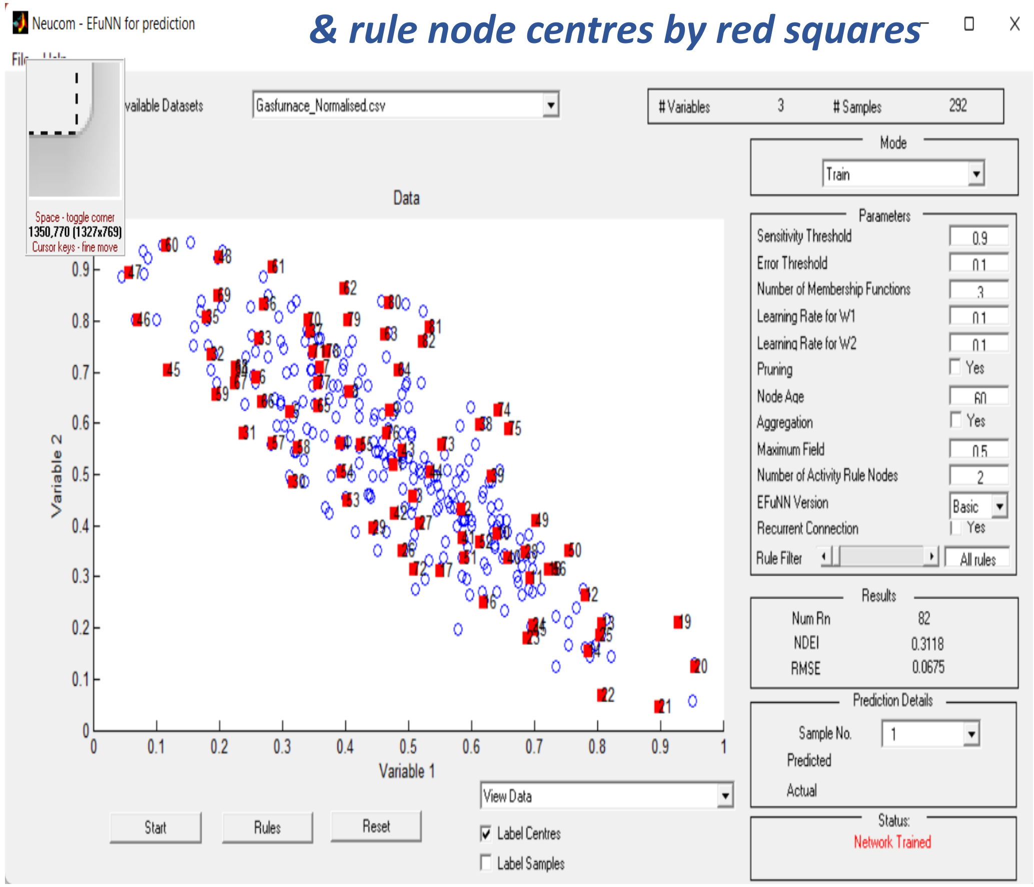
EFuNN Prediction Model: Parameters, Modelling and Analysis





EFuNN Prediction Model: Rules extraction

Data points are denoted as circles
& rule node centres by red squares



Rule 1:
if
[Var 1] --> (MF 1) @ 0.220 & (MF 2) @ 0.780 & (MF 3) @ 0.000 &
[Var 2] --> (MF 1) @ 0.000 & (MF 2) @ 0.806 & (MF 3) @ 0.194 &
then Output for (MF 1) @ 0.000 Output for (MF 2) @ 0.782 Output for (MF 3) @ 0.105

Rule 2:
if
[Var 1] --> (MF 1) @ 0.094 & (MF 2) @ 0.906 & (MF 3) @ 0.000 &
[Var 2] --> (MF 1) @ 0.000 & (MF 2) @ 0.835 & (MF 3) @ 0.165 &
then Output for (MF 1) @ 0.000 Output for (MF 2) @ 0.916

…
…
…

Rule 162:
if
[Var 1] --> (MF 1) @ 0.000 & (MF 2) @ 0.954 & (MF 3) @ 0.046 &
[Var 2] --> (MF 1) @ 0.000 & (MF 2) @ 0.481 & (MF 3) @ 0.519 &
then Output for (MF 1) @ 0.000 Output for (MF 2) @ 0.518 Output for (MF 3) @ 0.482

References:

1. N.Kasabov, *Time-Space, Spiking Neural Networks and Brain-Inspired Artificial Intelligence*, Springer, 2019.
2. N.Kasabov, *Evolving connectionist systems: Methods and Applications in Bioinformatics, Brain study and intelligent machines*, Springer Verlag, London, New York, Heidelberg, 2002.
3. N.Kasabov, *Foundations of neural networks, fuzzy systems and knowledge engineering*, MIT Press, CA, MA, 1996.
4. Q. Song and N. Kasabov, *NFI- A neuro-fuzzy inference method for transductive reasoning*, IEEE Tr. Fuzzy Systems, in print, 2004.
5. R. Fisher, *The use of Multiple Measurements in Taxonomic Problems*, Annals Eugenics 7, 1936.


Thank you!