Reverse Engineering Functional Brain Networks from fMRI Data Using Probabilistic Boolean Networks

Erin Boggess
Simpson College

Tiffany Jann
University of California, Berkeley

July 25, 2016

Abstract

This is the paper's abstract ...

Introduction

What we are trying to accomplish Why Probabilistic Boolean Networks? 1. We want Probabilistic Boolean Networks 2. Need to start from Boolean Networks to get PBN 3. What goes into creating Boolean Networks? Discretized Data and Static Partial Prior Knowledge Network. Therefore we want to get the best of both.

Goal: create a *method* that will reverse engineer functional brain networks from fMRI data using Probabilistic Boolean Networks. Steps: using a known (gold standard) network, generate data from that network using MULAN. Recall that we are starting with fMRI data and reverse-engineering probabilistic Boolean networks. Before we can obtain a PBN, we first need multiple Boolean networks that we can combine into a PBN. We obtain multiple Boolean networks from just one fMRI data set by using the sliding windows technique. To obtain any one dynamic Boolean network, we use the Reverse Engineering Algorithm with Evolutionary Computation Tools (REACT) developed by Dr. Vera-Licona in the paper x. This algorithm requires a binary time series input and allows a partial prior knowledge of the static network input. We decided to provide both inputs, in hopes of getting the best possible Boolean network. To obtain the binary time series, we discretized the original floating point fMRI data. To obtain a partial prior knowledge static network, we applied the 44 reverse engineering methods encoded into MULAN (42 neuroscience methods provided by the original author Dr. Wang and 2 molecular biology methods added by the REU students from 2015) to the fMRI data, found the top K performing methods, and finally combined their static network results into a consensus network by taking the average of the K networks. In verse order, we will explain each of these steps in the methods section.

The way to obtain a Probabilistic Boolean Network is by combining several Boolean Networks. To obtain several Boolean Networks

Methods

0.1 MULAN

To benchmark our methods, we used a MATLAB software toolbox called MULAN (MULtiple ANalysis Connectivity Toolbox). MULAN is able to simulate fMRI data from gold standard networks, infer networks from the data with 44 network inference methods, and evaluate the results. Though we used the MATLAB scripts in MULAN directly, it's also possible to use a GUI to interact with it. (REU 2015)

0.2 Gold Standard Networks

We selected two types of networks to simulate fMRI data from, Barabási–Albert scale free networks and Erdős–Rényi random networks. (REU 2015) We generated 200 scale free networks and 100 random networks, with 49 nodes in each network.

0.3 Generating Consensus Networks

abs value

We also wanted to optimize the partial prior knowledge static network input, leading us to the consensus network technique, "a 'wisdom of crowds' approach inspired from lessons learned by the molecular biology network inference community [4]."

We applied our fMRI data to the 44 network inference methods encoded in MULAN (42 well-accepted neuroscience methods and 2 top performing molecular biology methods from the DREAM5 Challenge) and identified the top performers by benchmarking methods using both ACC (accuracy) and AUC (area under the receiving operator characteristic (ROC) curve) metrics. We were able to obtain accuracy and AUC values by comparing the network inference method output static networks to the original network that we generated the data fMRI data from. This gave us the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). An example network inference method output is included in Table 1, accuracy is defined as $ACC = \frac{TP+TN}{TP+FP+TN+FN}$, and an example of an ROC curve is included in Figure 6.

0	0	0	0	0
0.125	0	0	0	0.125
0.125	0	0	0	0
0.125	0	0	0	0
0	0	0	0	0

Table 1: Example Gold Standard Network

0.992	0.9992	0.6568	0.9236	0.7648
0.1788	0.1312	0.134	0.1672	0.7884
0.5504	0.0912	0.9272	0.6452	0.1164
0.0788	0.5784	0.082	0.064	0.1304
0	0	0	0	0

Table 2: Example output of a network inference method

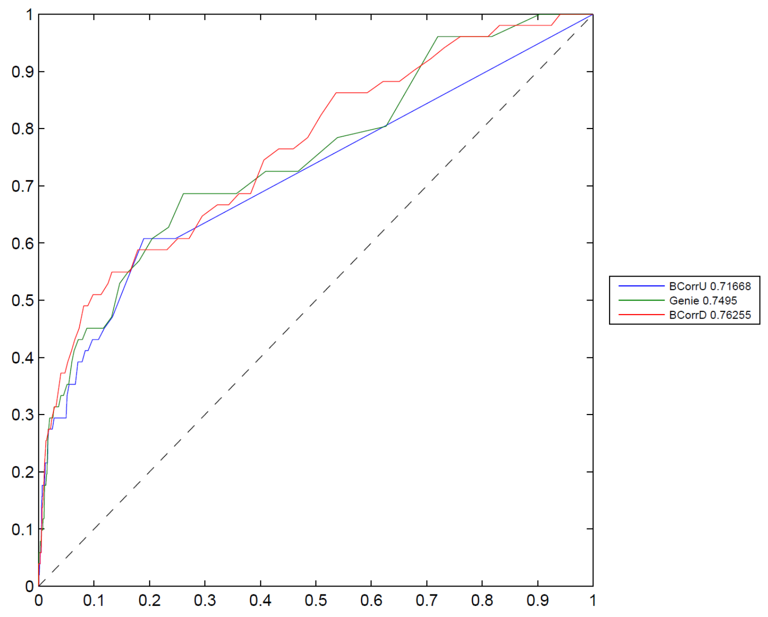


Figure 1: 3 Examples of ROC Curves

We took several weighted sums of the ACC and AUC metrics, as shown in Table 2. With each of these weighted sums, we identified our top performers and generated several consensus networks (one mean and one median consensus network for each cutoff from 3 to 10). After we have all 16 consensus networks for each of the weighted sums, we recomputed the top network inference methods, now including the consensus networks with the original 44 methods. This allowed us to compare the AUC and ACC values for each of these consensus networks and to the 44 original methods. We found that good combinations consistently outperformed all individual methods.

and applied the consensus network technique to the top performers. They formed consensusfound that consensus networks of the top 5 and 10 performers consistently outperformed any of its constituent methods.

0.4 Discretization

fMRI data is recorded in floating point values, while REACT requires Boolean input values, so we need to discretize our data into two states. To get the best possible Boolean network, we wanted the best possible input data, so we looked several dis-

Method	AUC	ACC	(0.1,0.9)	(0.2,0.8)	(0.3,0.7)	(0.4,0.6)	(0.5,0.5)	(0.6,0.4)	(0.7,0.3)	(0.8,0.2)	(0.9,0.1)
BCorrD	8601.0	8209.0	8248.2	8287.4	8326.6	8365.8	8405.0	8444.2	8483.4	8522.6	8561.8
Genie	8510.0	7916.0	7975.4	8034.8	8094.2	8153.6	8213.0	8272.4	8331.8	8391.2	8450.6
BCohF	8170.0	2631.0	3184.9	3738.8	4292.7	4846.6	5400.5	5954.4	6508.3	7062.2	7616.1
Tigress	7980.0	7727.0	7752.3	7777.6	7802.9	7828.2	7853.5	7878.8	7904.1	7929.4	7954.7
BCorrU	7968.0	2612.0	3147.6	3683.2	4218.8	4754.4	5290.0	5825.6	6361.2	6896.8	7432.4
PCohF	7688.0	2500.0	3018.8	3537.6	4056.4	4575.2	5094.0	5612.8	6131.6	6650.4	7169.2
BH2D	7432.0	5698.0	5871.4	6044.8	6218.2	6391.6	6565.0	6738.4	6911.8	7085.2	7258.6
BTED	7087.0	6254.0	6337.3	6420.6	6503.9	6587.2	6670.5	6753.8	6837.1	6920.4	7003.7
BH2U	7025.0	2364.0	2830.1	3296.2	3762.3	4228.4	4694.5	5160.6	5626.7	6092.8	6558.9
BTEU	6769.0	6258.0	6309.1	6360.2	6411.3	6462.4	6513.5	6564.6	6615.7	6666.8	6717.9
PTED	6690.0	6211.0	6258.9	6306.8	6354.7	6402.6	6450.5	6498.4	6546.3	6594.2	6642.1
PTEU	5988.0	6106.0	6094.2	6082.4	6070.6	6058.8	6047.0	6035.2	6023.4	6011.6	5999.8
BMITD2	5606.0	6238.0	6174.8	6111.6	6048.4	5985.2	5922.0	5858.8	5795.6	5732.4	5669.2
PMITD2	5585.0	6244.0	6178.1	6112.2	6046.3	5980.4	5914.5	5848.6	5782.7	5716.8	5650.9
PMITD1	5514.0	6231.0	6159.3	6087.6	6015.9	5944.2	5872.5	5800.8	5729.1	5657.4	5585.7
PH2U	5497.0	1795.0	2165.2	2535.4	2905.6	3275.8	3646.0	4016.2	4386.4	4756.6	5126.8
PH2D	5418.0	5574.0	5558.4	5542.8	5527.2	5511.6	5496.0	5480.4	5464.8	5449.2	5433.6
PCorrD	5240.0	5843.0	5782.7	5722.4	5662.1	5601.8	5541.5	5481.2	5420.9	5360.6	5300.3
BMITD1	4761.0	6055.0	5925.6	5796.2	5666.8	5537.4	5408.0	5278.6	5149.2	5019.8	4890.4
PMITU	4571.0	1705.0	1991.6	2278.2	2564.8	2851.4	3138.0	3424.6	3711.2	3997.8	4284.4
AS	4034.0	5616.0	5457.8	5299.6	5141.4	4983.2	4825.0	4666.8	4508.6	4350.4	4192.2
BMITU	3973.0	1713.0	1939.0	2165.0	2391.0	2617.0	2843.0	3069.0	3295.0	3521.0	3747.0
MVAR	3952.0	5732.0	5554.0	5376.0	5198.0	5020.0	4842.0	4664.0	4486.0	4308.0	4130.0
pCOH2	3853.0	1641.0	1862.2	2083.4	2304.6	2525.8	2747.0	2968.2	3189.4	3410.6	3631.8
Smvar	3702.0	1723.0	1920.9	2118.8	2316.7	2514.6	2712.5	2910.4	3108.3	3306.2	3504.1
hmvar	3628.0	5729.0	5518.9	5308.8	5098.7	4888.6	4678.5	4468.4	4258.3	4048.2	3838.1
dDTF	3619.0	5722.0	5511.7	5301.4	5091.1	4880.8	4670.5	4460.2	4249.9	4039.6	3829.3
PDC	3361.0	5699.0	5465.2	5231.4	4997.6	4763.8	4530.0	4296.2	4062.4	3828.6	3594.8
Af	3206.0	5688.0	5439.8	5191.6	4943.4	4695.2	4447.0	4198.8	3950.6	3702.4	3454.2
oPDCF	3193.0	5676.0	5427.7	5179.4	4931.1	4682.8	4434.5	4186.2	3937.9	3689.6	3441.3
COH1	3190.0	1564.0	1726.6	1889.2	2051.8	2214.4	2377.0	2539.6	2702.2	2864.8	3027.4
COH2	3157.0	1604.0	1759.3	1914.6	2069.9	2225.2	2380.5	2535.8	2691.1	2846.4	3001.7
GC	2663.0	5621.0	5325.2	5029.4	4733.6	4437.8	4142.0	3846.2	3550.4	3254.6	2958.8
PCorrU	2529.0	1541.0	1639.8	1738.6	1837.4	1936.2	2035.0	2133.8	2232.6	2331.4	2430.2
DC1	2528.0	5653.0	5340.5	5028.0	4715.5	4403.0	4090.5	3778.0	3465.5	3153.0	2840.5
CondGC	2497.0	5636.0	5322.1	5008.2	4694.3	4380.4	4066.5	3752.6	3438.7	3124.8	2810.9
DTF	2423.0	5582.0	5266.1	4950.2	4634.3	4318.4	4002.5	3686.6	3370.7	3054.8	2738.9
ffDTF	2423.0	5582.0	5266.1	4950.2	4634.3	4318.4	4002.5	3686.6	3370.7	3054.8	2738.9
GGC	2344.0	5651.0	5320.3	4989.6	4658.9	4328.2	3997.5	3666.8	3336.1	3005.4	2674.7
PGC	2317.0	5601.0	5272.6	4944.2	4615.8	4287.4	3959.0	3630.6	3302.2	2973.8	2645.4
GPDC	2310.0	5614.0	5283.6	4953.2	4622.8	4292.4	3962.0	3631.6	3301.2	2970.8	2640.4
BCohW	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0
PCohW	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0
pCOH1	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0	400.0

Table 3: Different weighted sums of (AUC,ACC) metrics.

cretization techniques. Discretization refers to the reduction of raw data into discrete values, and the way each discretization technique achieves this varies from using metric cutoffs, ranking cutoffs, clustering, and variation between time points (Gallo *et. al.*) Since Booleanizing 204 time point fMRI data requires unsupervised binary discretization techniques fit for long time series, we eliminated 9 methods out of the 20 discussed in “Discretization of gene expression data revised” by Cristian A. Gallo *et. al.* Since we discretized to only two levels, median, equal frequency, and Top $X\%$ (when $X = 50$) discretization techniques converged to one method. In the end, we had 9 fMRI data-friendly discretization techniques to rank and choose from: Mean, Median (also Top 50%, Equal Frequency Discretization), Max – $X\%$ Max, Equal Width Discretization, Gallo, k-means, bik-means, transitional state discrimination, and Erdal *et. al.*'s method.

0.4.1 Variable Inputs

We kept the input Level of Discretization at 2 to obtain binary values. For Data Scope (the context used to compute discretized values), mean, median, Max – X% Max, Equal Width, and k-means gave users the options: Row, Column, Matrix, while Gallo, Bik-means, TSD, and Erdal did not allow specification of scope, instead defaulting to Row, Row and Column, Data Point, Data Point, respectively. For methods with adjustable scope, we chose row, as reverse-engineering functional brain networks has strong parallels to inferring gene regulatory networks from association rules, from which "the most common approach is to use the gene expression profile to determine the discrete states of a gene g_i" (Gallo *et. al.*) COMMENT: He cites himself (Gallo et. al. 2013) for this. We should probably look into this.

Lastly, Top X%, Max − X% Max, and Erdal's each had one extra parameter: percent X, percent X, and threshold t, respectively. We found these parameters to be troublesome–tailoring the parameter would improve the performance of a discretization technique for one data set, but no parameter would generalize across data sets significantly better than the default values of 50, 50, and 1, respectively. We left the parameters at that, with Erdal's and Max − X% Max consistently under-performing regardless of the parameter.

0.4.2 Applying Discretization Techniques to fMRI Data

Gallo *et. al.* encoded 16 discretization methods into the software GED PRO TOOLS included with their paper. Using this applet, we applied the 9 suitable discretization techniques on a total of 20 data sets: 5 for each of the 4 different types of data described in the Obtaining fMRI Data section: ADHD patient data, control non-patient data, Barabási–Albert scale free network generated data, and Erdős–Rényi random network generated data.

0.4.3 Method for Benchmarking Discretization Techniques

To our knowledge, there is currently no systematic way to compare discretization methods, so to find the best methods for Booleanizing our data, we proposed a benchmarking method that directly compares the discretized output with the original data by the areas under each curve.

First, we pre-processed (normalized) the original fMRI data (Figure 1) by shifting all data points until the minimum was at 0 (Figure 2), then scaling until the maximum was at 1 (Figure 3). For one, this ensured that we would be dealing with positive areas, so that positive and negative areas do not cancel out. Normalizing the data also ensures that the discretized data has equal probability to over-estimate and under-estimate the original data.

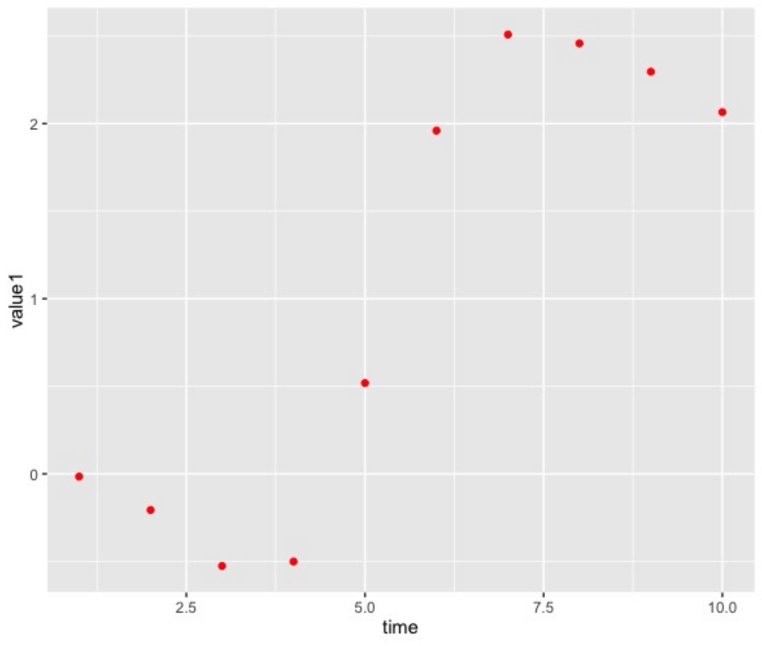


Figure 2: Raw Data

With the original data set normalized, we can apply our benchmarking algorithm to all 9 discretization results stemming from that data set. The code for this benchmarking method itself does not use plots, but rather relies on vectorized operations and functions. A heavily commented copy is available in the appendix, but here we will describe the concepts behind the

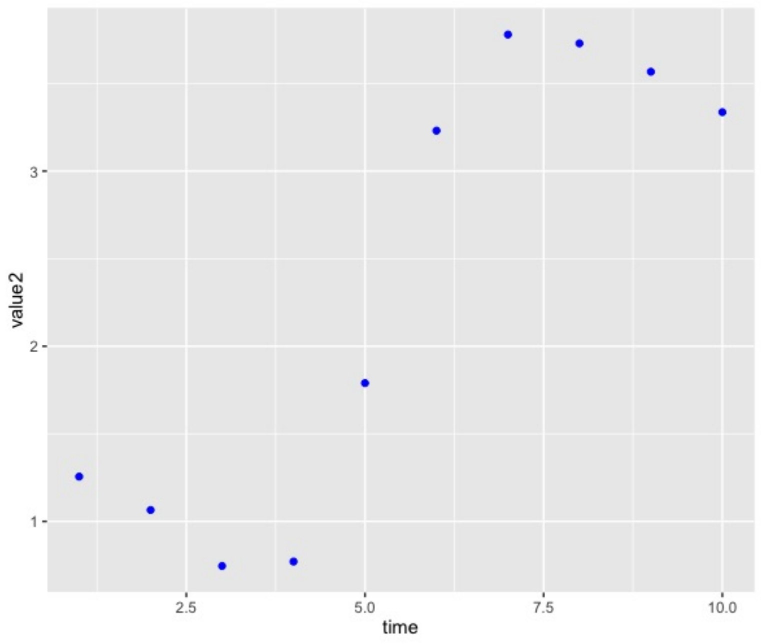


Figure 3: Shifted Data

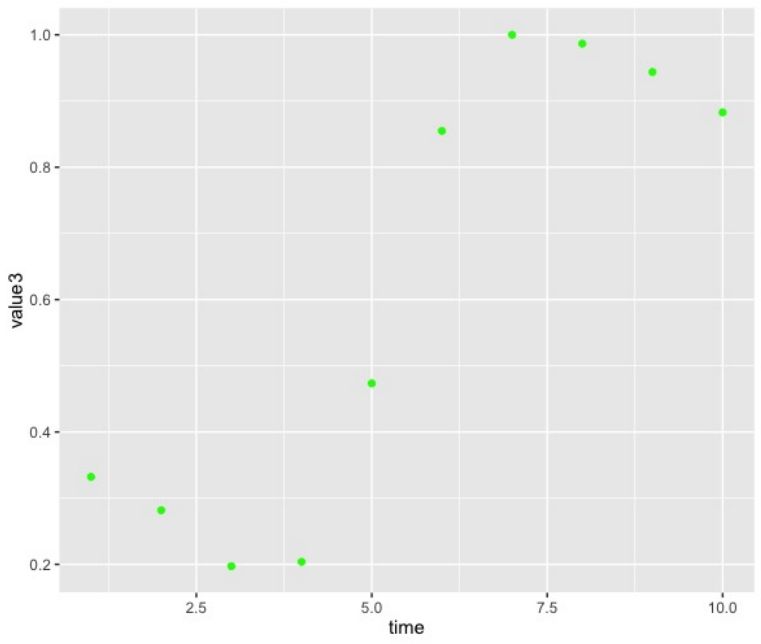


Figure 4: Shifted and Scaled Data

method.
As an example, we selected one patient data set, from which we selected one discretization method's output, from which we selected one node, from which we selected just 10 time points. We plot this with 10 time points of one node of the normalized original data (Figure 4). The discretized data points are in blue while the normalized original data points are in red. We then interpolated the linear piecewise-linear curve for our two time series in order to consider the area under the curves. Note that blue points are only connected to blue points, and likewise for the discretized data. To determine how well the discretization technique models the original data, we calculated the absolute area between the curves. In Figure 4, those "area errors" would be the red and blue. The magenta, or overlapping areas, are "correct areas."
In order to calculate these areas, we looked at time intervals of size 1. This gave the endpoints of the line segments whose equations we need to integrate the absolute value difference of the two functions, effectively giving us the absolute area between two line segments for that time interval. To find the area difference for one node, repeat this process for all 203 time intervals and sum them up. To find the area difference for one method, find the area for all 49 nodes and sum them up. Repeat for all methods corresponding to the same data set and obtain a total difference in area for each dicretization

technique for the specified data set. We repeated this entire process over our 20 data sets, as mentioned above, and found our top performers: mean, k-means, median, bik-means, and equal width (Figure 5).

For one time interval:

$$\int_{t}^{t+1} |(m_1x+b_1)-(m_2x+b_2)| \cdot dx$$

For one node (s time points):

$$\sum_{t=1}^{s-1}\int_{t}^{t+1} |(m_1x+b_1)-(m_2x+b_2)| \cdot dx$$

For one method (n brain regions):

$$\sum_{j=1}^{n}\sum_{t=1}^{s-1}\int_{t}^{t+1} |(m_1x+b_1)-(m_2x+b_2)| \cdot dx$$

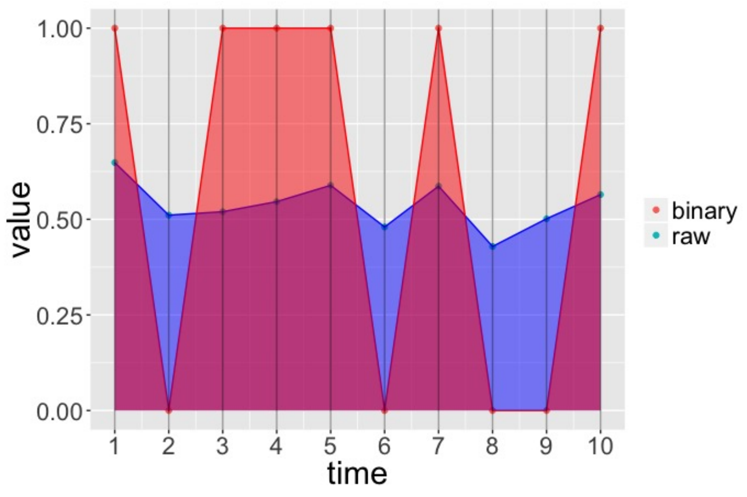
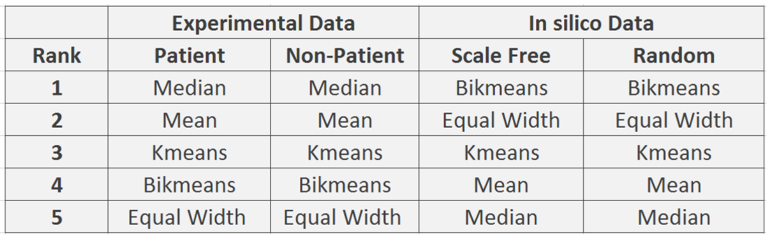


Figure 5: Discretized Data and Normalized Original Data



	Experimental Data		In silico Data	
Rank	Patient	Non-Patient	Scale Free	Random
1	Median	Median	Bikmeans	Bikmeans
2	Mean	Mean	Equal Width	Equal Width
3	Kmeans	Kmeans	Kmeans	Kmeans
4	Bikmeans	Bikmeans	Mean	Mean
5	Equal Width	Equal Width	Median	Median

Figure 6: Results: Top Performing Discretization Techniques

0.4.4 Summary of Top-Performing Discretization Techniques

Mean Discretizing by mean simply calculates the mean of the specified scope, and then for each value in that scope, if it is greater than or equal to the mean, that value is assigned to a 1, otherwise, 0.

Median Similar to discretizing by mean, calculate the median of the specified scope, and then for each value in that scope, if it is greater than or equal to the median, that value is assigned to a 1, otherwise, 0.

Interval Interval will take the maximum and minimum values of the scope, and take the average. That point is the cut-off, where every value in the scope greater than or equal to the cut-off is a 1, else 0.

k-means k-means clustering is among the most widely known unsupervised learning techniques. Note that the GED PRO TOOLS implementation has a set process to determine initial cluster centers, rather than running several simulations with random initial cluster centers and settling with recurring clusters.

bik-means Bik-means is detailed in the paper by Yong Li, *et. al.*

References

[1] D. Adams. *The Hitchhiker's Guide to the Galaxy.* San Val, 1995.