Graph models for machine learning: K-associated graphs and Attribute-based Decision Graphs

João R. Bertini Jr.

School of Technology
University of Campinas, SP, Brazil

bertini@ft.unicamp.br

August 16, 2018

Outline

Motivation - Why graphs?

- Graphs have been successfully employed in plenty of supervised, unsupervised and semi-supervised tasks.
- Traditional methods provides:
  - Representing arbitrary distribution
  - Uncover topological relationships
  - Allow Hierarchical representation
  - High computational costs

The **K-associated graph** and the **Attribute-based Decision graphs**

- Alternative graph representations
- Low computational costs
- Intuitive and probabilistic representation of the data

The K-associated Graph

The K-associated graph

- Consider a data set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ where each case $\mathbf{x}_i$ has an associated class label $c_i \in \Omega = \{\omega_1, \omega_2, ..., \omega_M\}$.
- The K-associated graph is constructed as follows:[1].
  1. Each $\mathbf{x}_i$, is represented by a vertex v_i, accordingly c_i stands for the label of v_i.
  2. Connect v_i to all its K nearest neighbors that share the same class.

  Properties
  - The K-associated graph can be seen as a set of dinjoint components $C_\alpha \in C = \{C_1, ..., C_R\}$ $N \leq R \leq M$.
  - As only vertices belonging to the same class can be connected. Each component is associated to only one data class.
  - Raising K, monotonically decreases the number of components.

[1] J.R. Bertini Jr., L. Zhao, R. Motta, A.A. Lopes, A nonparametric classification method based on k-associated graphs, Information Sciences 181 (2011) 5435–5456.

The Purity measure in the K-associated Graph

- Let the D_α be the average degree of component C_α
  - and degree of v_i being $d_i = d_i^{in} + d_i^{out}$
- And the **purity measure** of the component C_α, denoted as Φ_α, is defined by $\Phi_\alpha = \frac{D_\alpha}{2K}$.

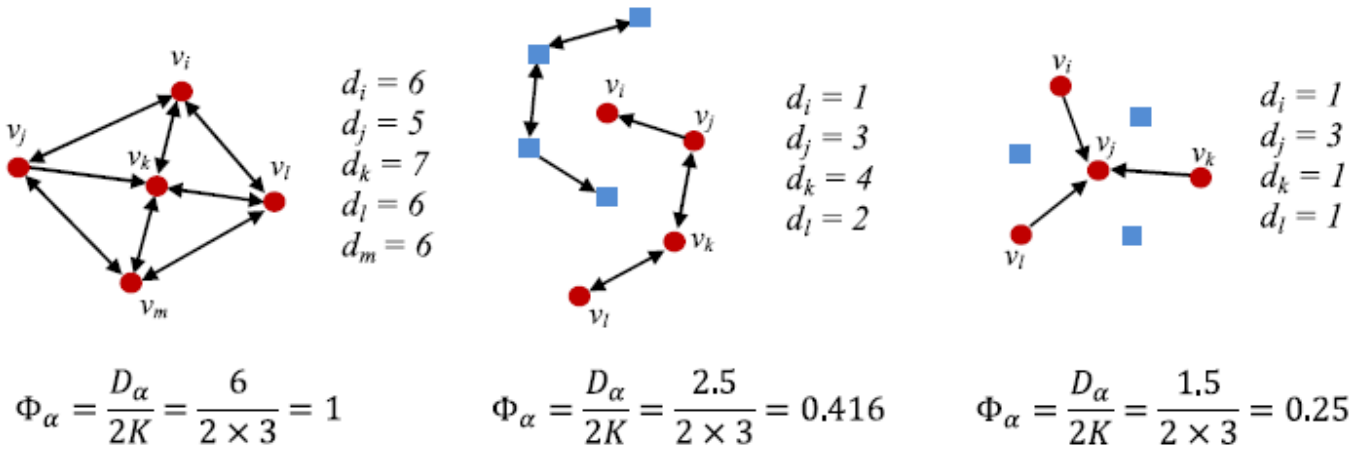
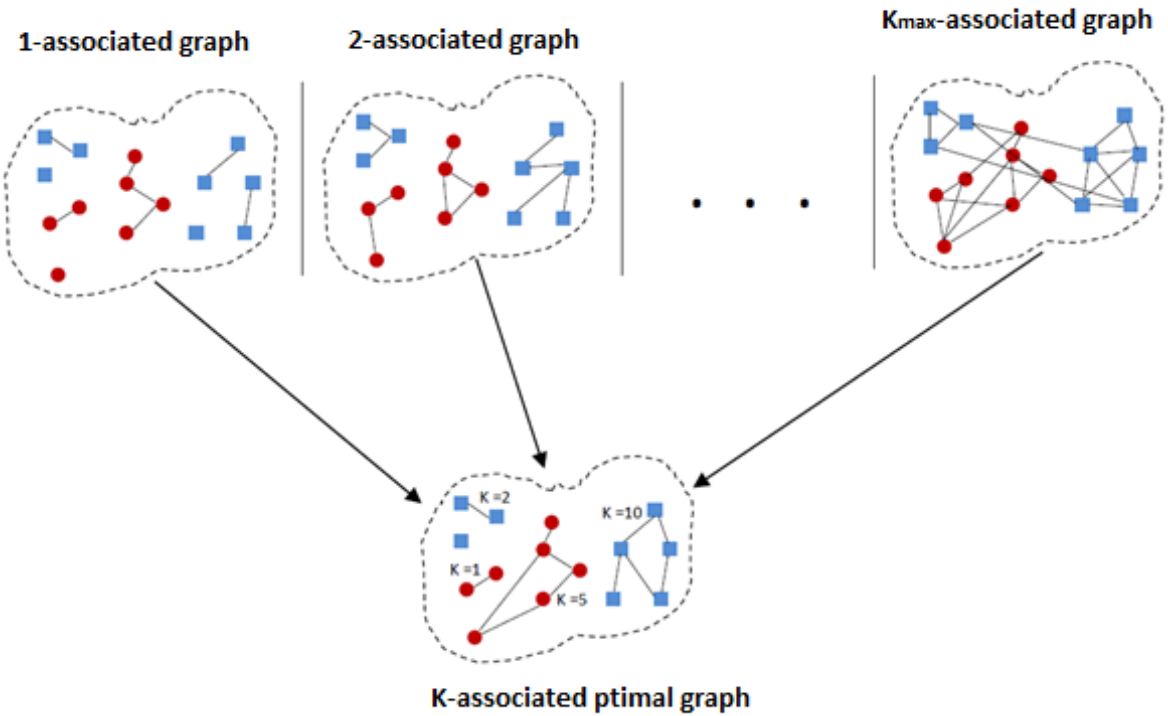


Figure: Examples of purity calculation considering $K = 3$.

- Purity quantifies how intertwined are the vertices of different components.
- Can be used as *a priori* probability for the component.

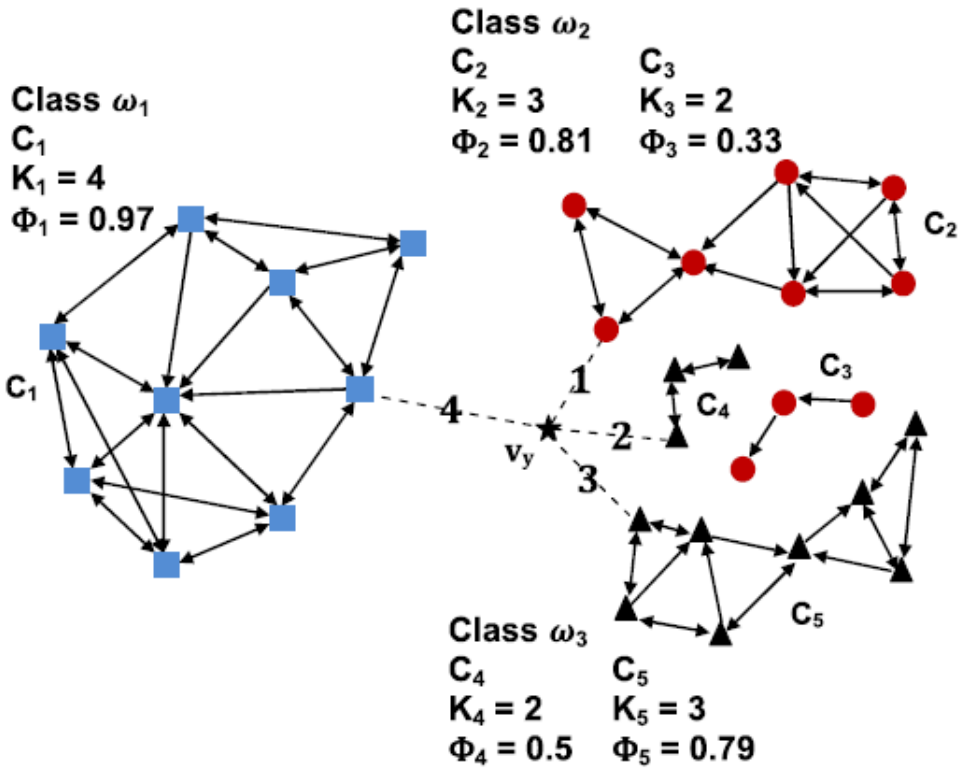
The K-associated optimal graph

- By varying K, some graphs may have better components than others according to the purity measure.
- We want to obtain a graph with the best components!
  - Increase K, while keeping track of the best components
  - By replacing components in the optimal graph according to: $\Phi_{\beta}^{(K+z)} \geq \Phi_{\alpha}^{(K)}$ for all $C_{\alpha}^{(K)} \subseteq C_{\beta}^{(K+z)}$



The KAOG classifier

- During classification, purity and K of each component are used to infer the probability of new case to belong to it.



Classification of stationary data

KAOG - some classification results

Comparison results through **fifteen** knowledge domains each with **three levels of noise**. All algorithm used for comparison had its paramter adjusted in the model selection phase: KNN and Weighted KNN (k), Prototype KNN (p), C4.5 (cf,m) and M-SVM (C,c).

Table: Average accuracy rank of each algorithm over the 15 data sets

	KAOG	***K*NN**	**Weight. *K*NN**	***K*NN Prot.**	**C4.5**	**M-SVM**
Avg. rank orig. data	2.63	3.50	2.46	5.40	4.26	2.66
Avg. rank noisy data	2.93	3.20	3.00	5.30	3.63	2.90
Avg. rank all data	2.83	3.30	2.82	5.35	3.84	2.82

KAOG - some classification results

Table: Classification results for the algorithms KAOG, *K*NN, Weighted *K*NN and Gibbs Sampling when using the HEOM similarity measure

Domain	**KAOG**	***K*NN**	**Weighted *K*NN**	**Gibbs**
		HEOM		
Acute	**99.3±2.3** (1)	94.5±8.5 (k=1)(4)	95.4±8.3 (k=1)(3)	96.9±4.8 (k=1)(2)
Heart	74.1±7.9 (4)	76.0±3.06 (k=7)(2)	**76.3±3.8** (k=10)(1)	74.6±7.8 (k=2)(3)
Soybean	90.7±3.0 (2)	75.6±3.4 (k=1)(4)	75.9±3.5 (k=2)(3)	**91.1±2.8** (k=3)(1)
Dermatology	**94.5±3.3** (1)	86.4±3.35 (k=1)(4)	86.6±3.2 (k=2)(3)	94.1±3.0 (k=5)(2)
Horse	69.9±6.1 (2)	67.1± 2.7 (k=9)(4)	68.0±2.8 (k=10)(3)	**74.7±6.7** (k=1)(1)
Voting	**92.9±3.7** (1)	90.5±1.6 (k=3)(4)	90.7±1.5 (k=4)(3)	92.6±3.6 (k=6)(2)
Mammography	74.4±3.8 (4)	**81.7±0.72** (k=19)(1)	80.0±1.4 (k=29)(3)	81.4±4.2 (k=9)(2)
Audiology	63.1±9.5 (2)	44.7±5.2 (k=1)(4)	45.2±5.1 (k=2)(3)	**67.2±9.3** (k=3)(1)
CTG	99.3±0.4 (2)	97.8±0.6 (k=1)(4)	98.2±0.8 (k=3)(3)	**99.6±0.4** (k=2)(1)
Annealing	**98.9±2.1** (1)	88.9±2.2 (k=1)(4)	89.1±2.4 (k=4)(3)	97.2±1.7 (k=2)(2)
Average Rank	1.9	3.5	3.8	1.8

KAOG - some classification results

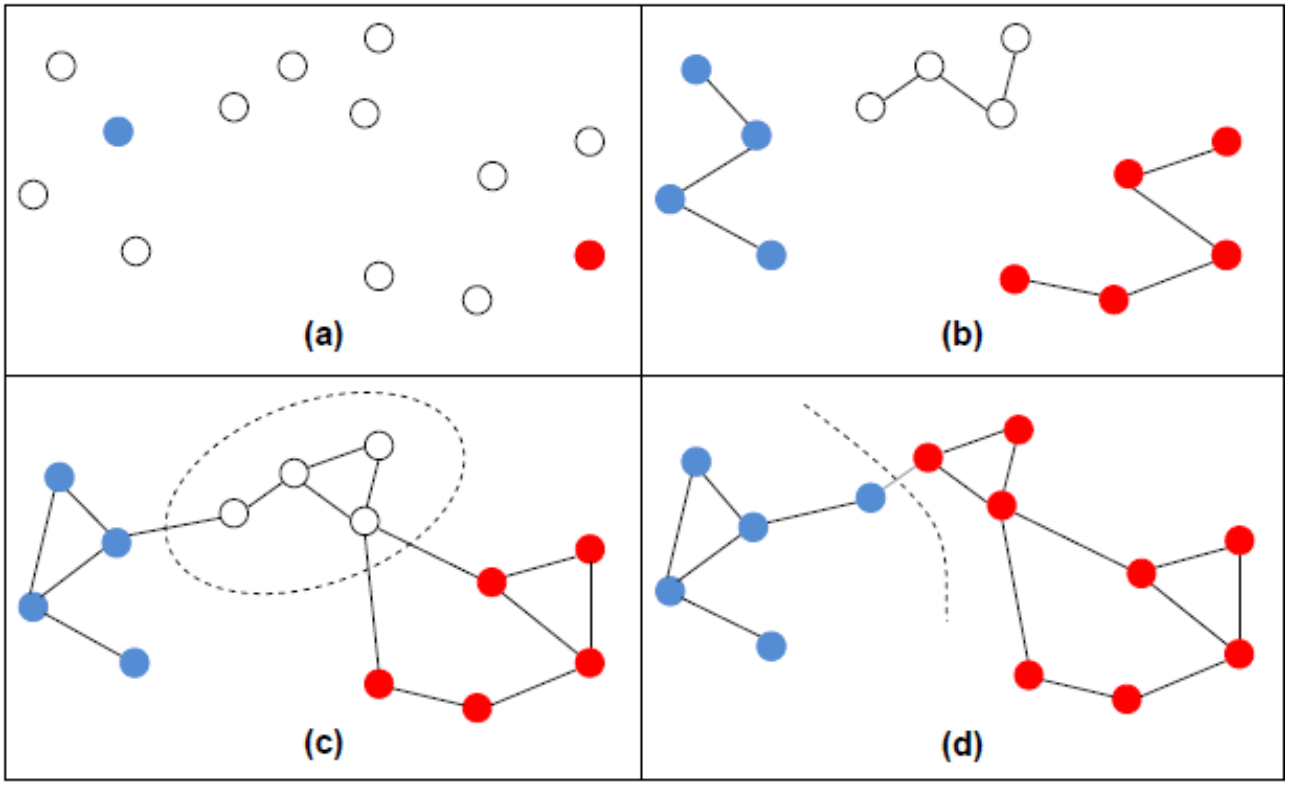
Table: Classification results for the algorithms KAOG, *K*NN, Weighted *K*NN and Gibbs Sampling when using the HVDM similarity measure

Domain	**KAOG**	***K*NN**	**Weighted *K*NN**	**Gibbs**
		HVDM		
Acute	**100.0±0.0** (1)	96.0±7.6 (k=1)(3.5)	96.0±7.7 (k=1)(3.5)	97.2±4.6 (k=1)(2)
Heart	78.4±6.4 (4)	81.1±3.0 (k=7)(2)	**81.2±2.9** (k=8)(1)	80.4±7.6 (k=2)(3)
Soybean	**91.5±2.9** (1)	81.8±3.7 (k=1)(4)	81.9±3.7 (k=2)(3)	90.2±2.6 (k=2)(2)
Dermatology	**97.6±2.3** (1)	94.2±3.0 (k=1)(3.5)	94.2±3.1 (k=1)(3.5)	95.8±1.8 (k=3)(2)
Horse	**99.6±0.94** (1.5)	99.5± 0.9 (k=3)(3)	**99.6±3.1** (k=4)(1.5)	98.5±1.8 (k=1)(4)
Voting	**95.6±3.0** (1)	93.3±1.8 (k=1)(4)	93.6±1.8 (k=2)(3)	94.9±3.1 (k=6)(2)
Mammography	74.3±4.7 (4)	**81.9±1.1** (k=9)(1)	81.3±0.9 (k=27)(3)	81.7±3.1 (k=9)(2)
Audiology	71.5±8.1 (2)	52.2±6.4(k=1)(4)	52.3±6.8 (k=1)(3)	**79.0±8.9** (k=1)(1)
CTG	**99.8±0.2** (1.5)	99.7±6.4 (k=1)(3.5)	99.7±0.2 (k=1)(3.5)	**99.8±0.2** (k=2)(1.5)
Annealing	93.1±2.4 (2)	87.5±2.3 (k=2)(3)	87.3±1.8 (k=12)(4)	**94.3±2.2** (k=1)(1)
Average Rank	1.9	3.15	2.9	2.05

Semi-supervised learning (transduction)

Semi-supervised learning (transduction)

- Transductive learning is a branch of semi-supervised learning concerned with the task of spreading labels on a finite set of data.
- KAOGSS[2] is the K-associated variant for transductive learning



[2] Bertini JR., J. R.; Zhao, L. A Purity Measure Based Transductive Learning Algorithm. Lecture Notes in Computer Science. Springer, 2013. v. 7952. p. 405-412.

Semi-supervised learning (transduction)

Table: Comparison results considering 100 labeled patterns.

Algorithm	g241c	g241d	Digit1	USPS	COIL	BCI	Text
KAOGSS	43.13	43.50	1.94	5.77	9.98	34.12	25.52
1-NN	43.93	42.45	3.89	5.81	17.35	48.67	30.11
SVM	23.11	24.64	5.53	9.75	22.93	34.31	26.45
MVU + 1-NN	43.01	38.20	2.83	6.50	28.71	47.89	32.83
LEM + 1-NN	40.28	37.49	6.12	7.64	23.27	44.83	30.77
QC + CMN	22.05	28.20	3.15	6.36	10.03	46.22	25.71
Discrete Reg.	43.65	41.65	2.77	4.68	9.61	47.67	24.00
TSVM	18.46	22.42	6.15	9.77	25.80	33.25	24.52
SGT	17.41	9.11	2.61	6.80	-–	45.03	23.09
Cluster-Kernel	13.49	4.95	3.79	9.68	21.99	35.17	24.38
Data-Dep. Reg.	20.31	32.82	2.44	5.10	11.46	47.47	-–
LDS	18.04	23.74	3.46	4.96	13.72	43.97	23.15
Laplacian RLS	24.36	26.46	2.92	4.68	11.92	31.36	23.57
CHM (normed)	24.82	25.67	3.79	7.65	-–	36.03	–

Classification of non-stationary data

Classification in data stream

The KAOGINC[3] is the incremental version of the KAOG algorithm to cope with non-stationary data streams.

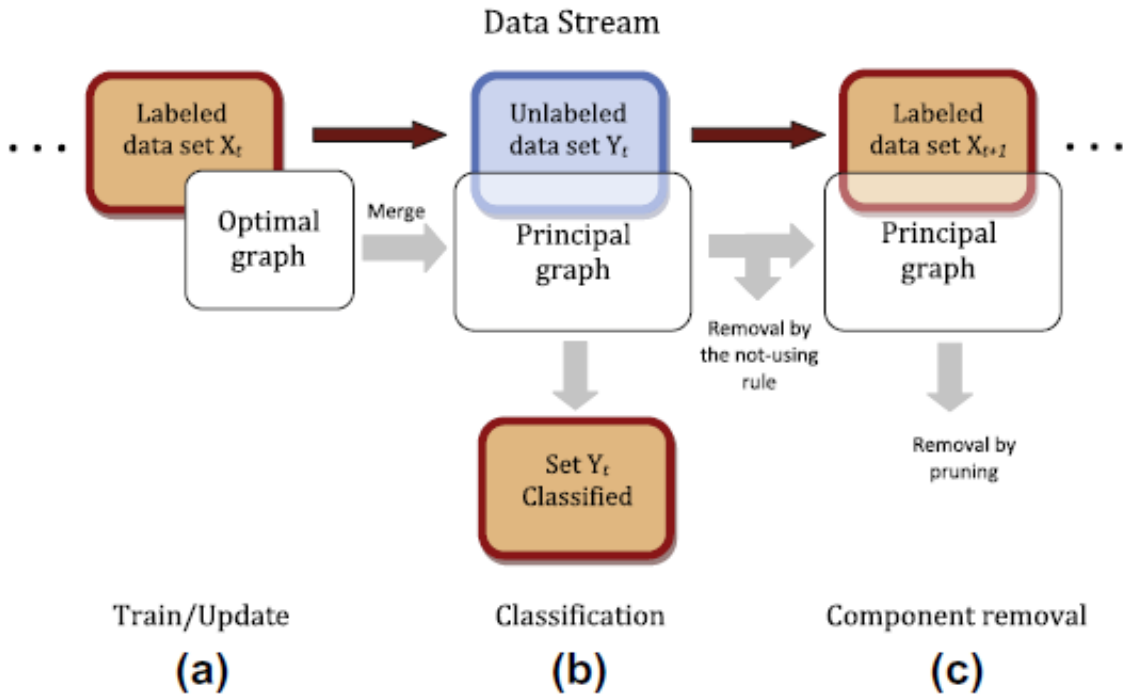
- In non-stationary classification tasks, the underlying data distribution changes over time.
- Ideally, incremental learning algorithms must provide:
  - Good classification performance
  - Stability
  - Low computational costs (processing time and memory)

- A data stream can be addressed as a sequence of data chuncks $S = \{Y_1, X_1, \ldots, Y_t, X_t, \ldots\}$,
  - X_t stands for labeled data sets
  - Y_t are unlabeled data sets

[3] J.R. Bertini Jr., L. Zhao, A. A. Lopes. An incremental learning algorithm based on the K-associated graph for non-stationary data classification. Information Sciences, v. 246, p. 52-68, 2013.

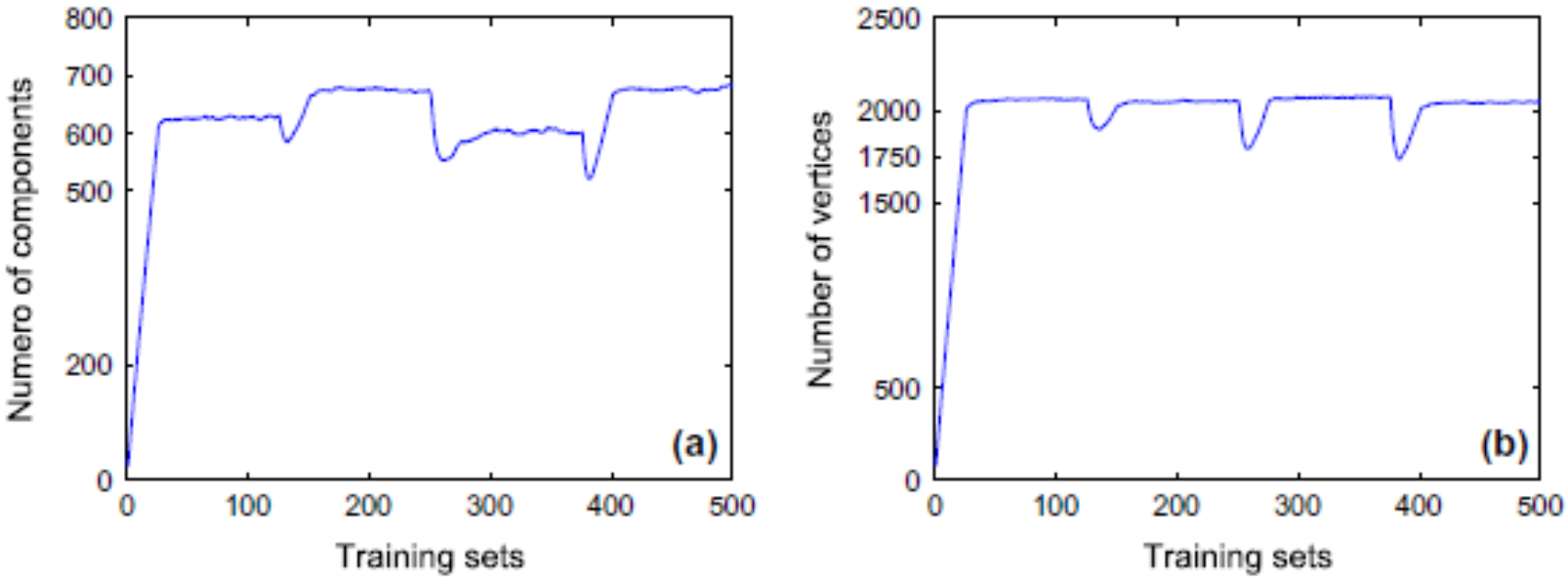
The KAOGINC algorithm

- The KAOGINC scheme for data stream processing.



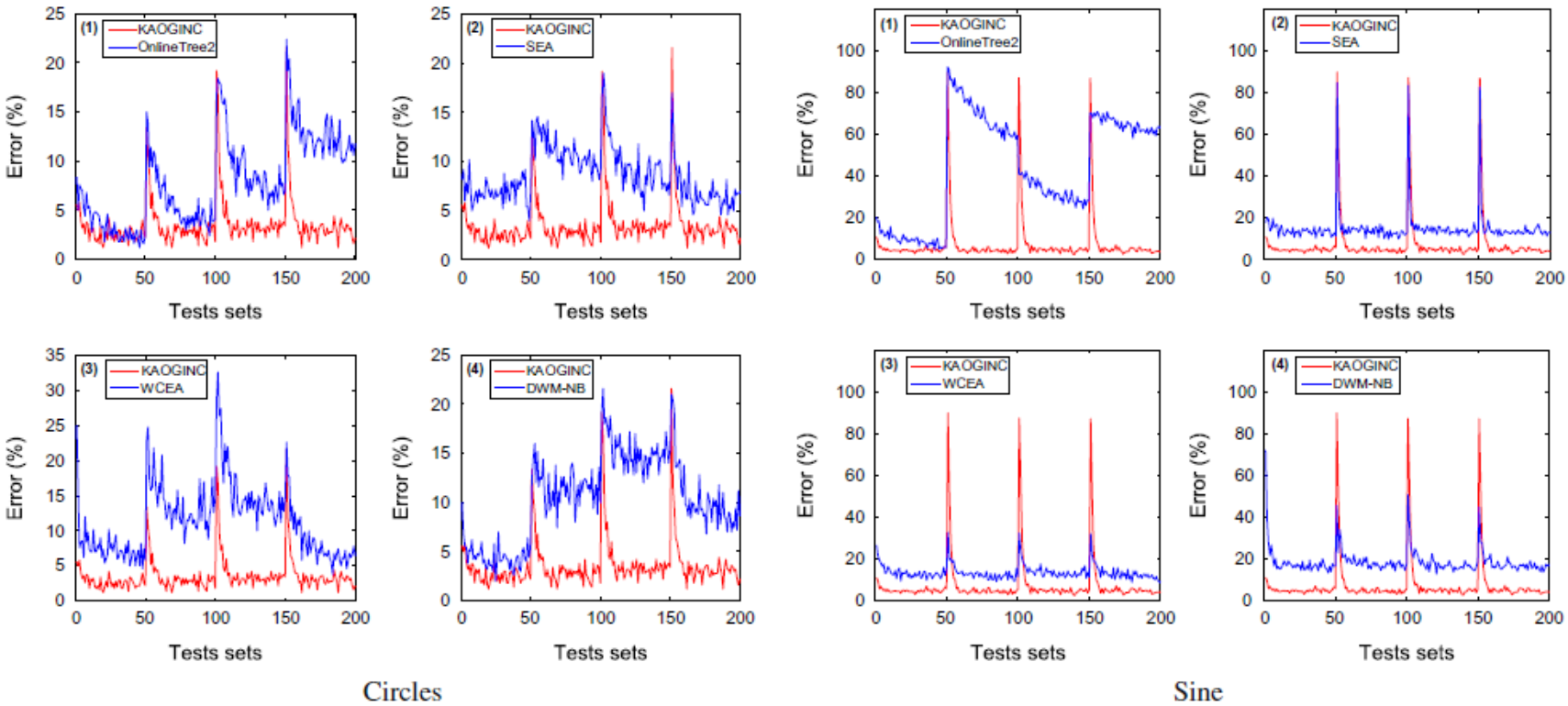
The KAOGINC algorithm

- KAOGINC stability and component structure.



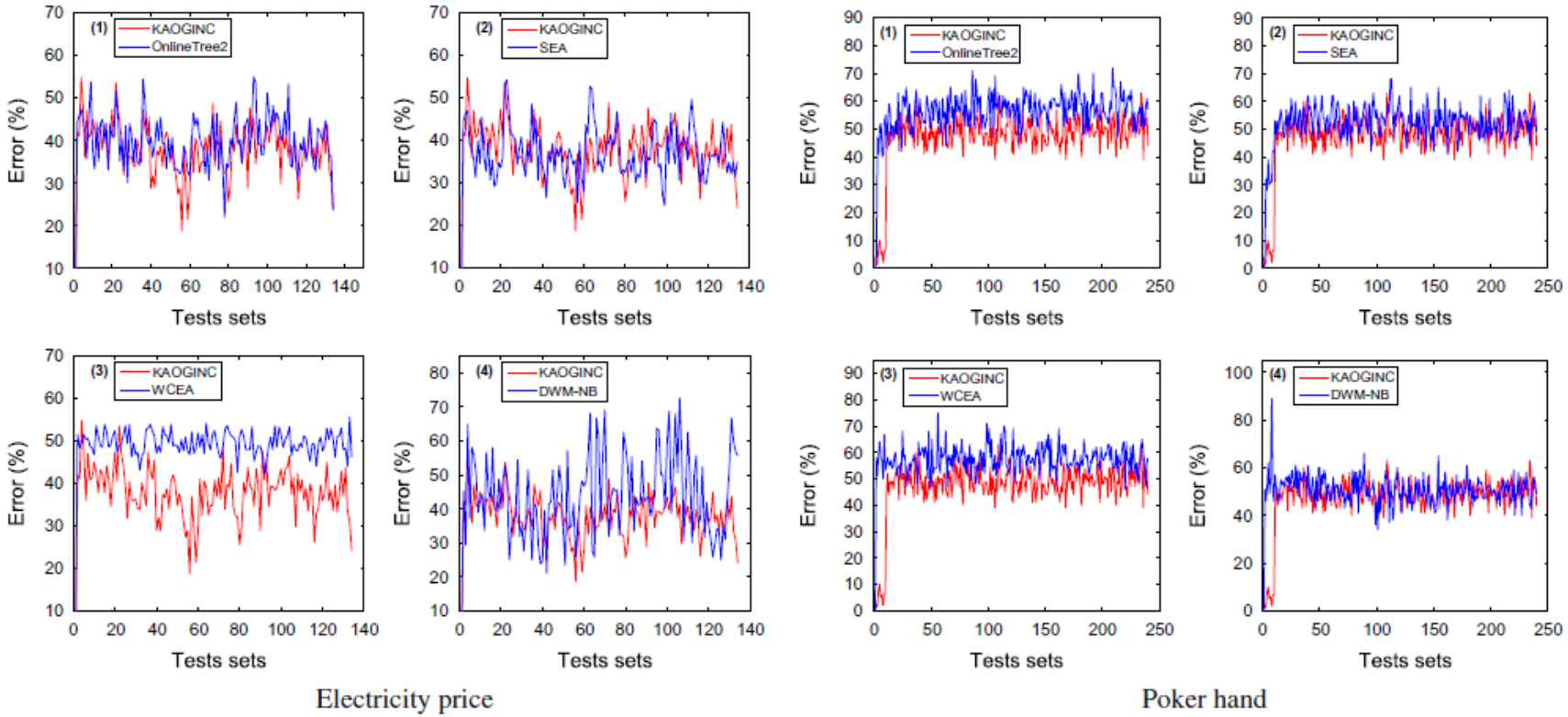
Experiments results

- Error percentage along the stream processing - Artificial domains Circle and Sine.



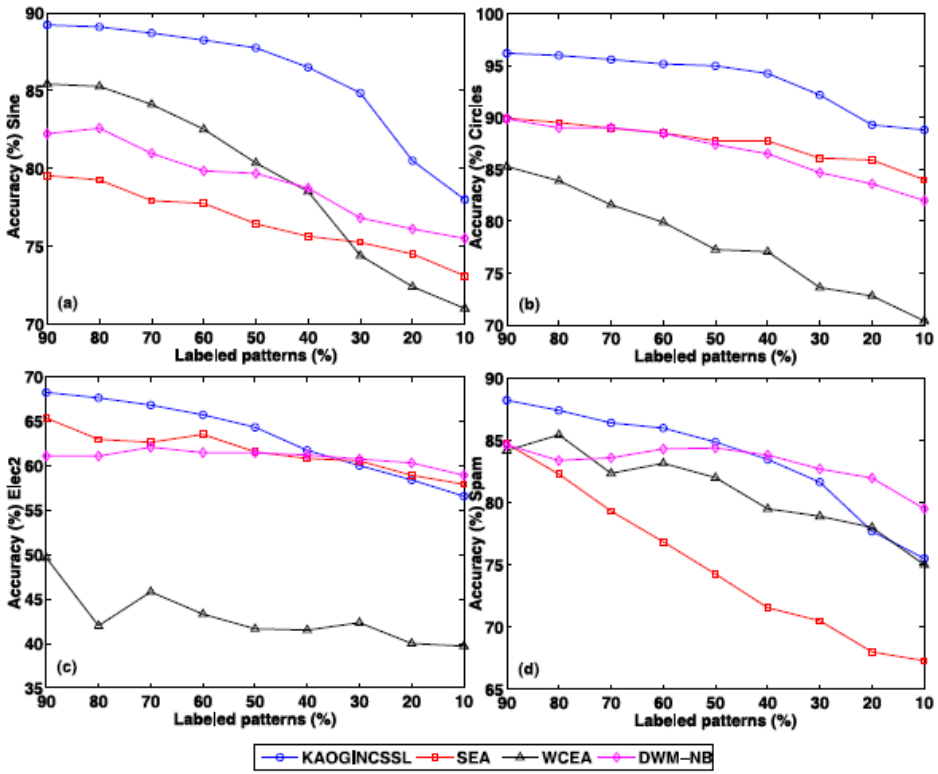
Experiments results

- Error percentage along the stream processing - Real domains Electricity price and Poker hand.



Experiments results - semi-supervised learning

- Data stream classification with **partially labeled data**.
- KAOGINC (incremental) + KAOGSS (transduction) = KAOGINCSSL[4]

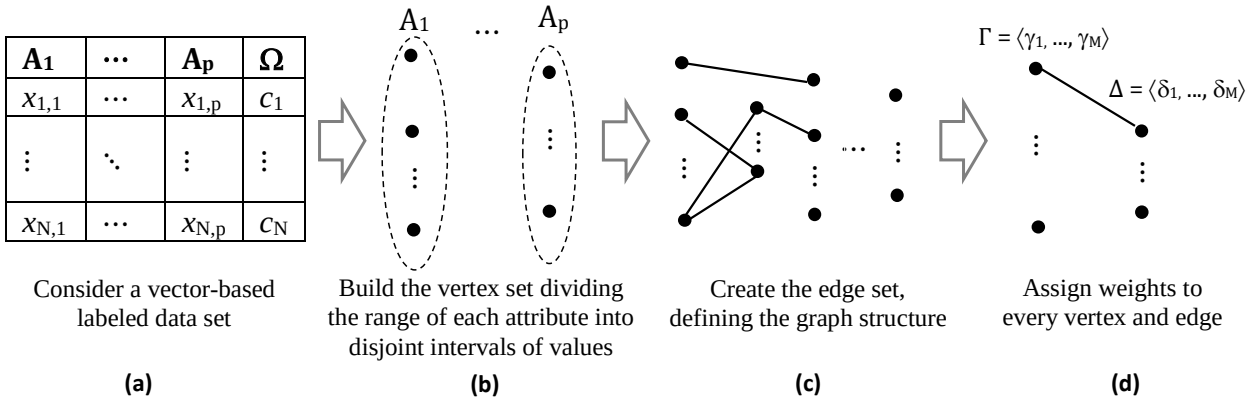


[4] Bertini, JR. J.R. Lopes, A., Zhao, L. Partially labeled data stream classification with the semi-supervised K-associated graph. Journal of The Brazilian Computer Society, v. 18, p. 1-12, 2012.

Attribute-based Decision Graphs (AbDG)

Attribute-based Decision Graphs

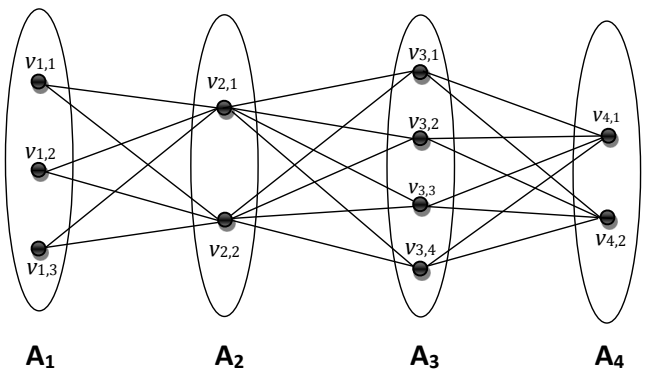
- *The Attribute based Decision Graph (AbDG) is built from a given data set aiming at mapping its attribute values interrelations to a graph structure.*[5]
  - A **vertex** represents an **interval of values** within an attribute;
  - **Edges** are established between **vertices from different attributes** accordingly to their values



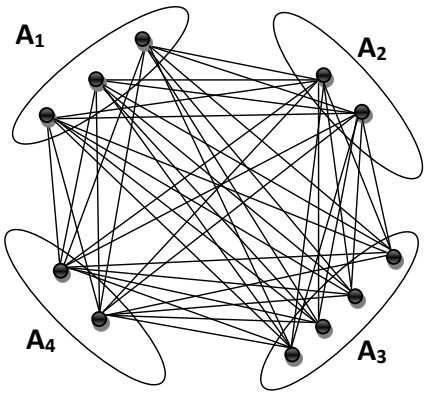
[5] J. R. Bertini Jr., M. C. Nicoletti, and Z. Liang, "Attribute-based decision graphs: a framework for multiclass data classification," Neural Networks, vol. 85, pp. 69–84, 2017.

Attribute-based Decision Graphs

- Possible graph structures
  - (a) **p-partite** graph - set of vertices are connected according to a given order
    - order interferes with graph structure
    - provides a more concise representation
  - (b) **complete p-partite** graph - every pair of set of vertices is connected
    - order does not interferes with graph structure
    - provide complex representation



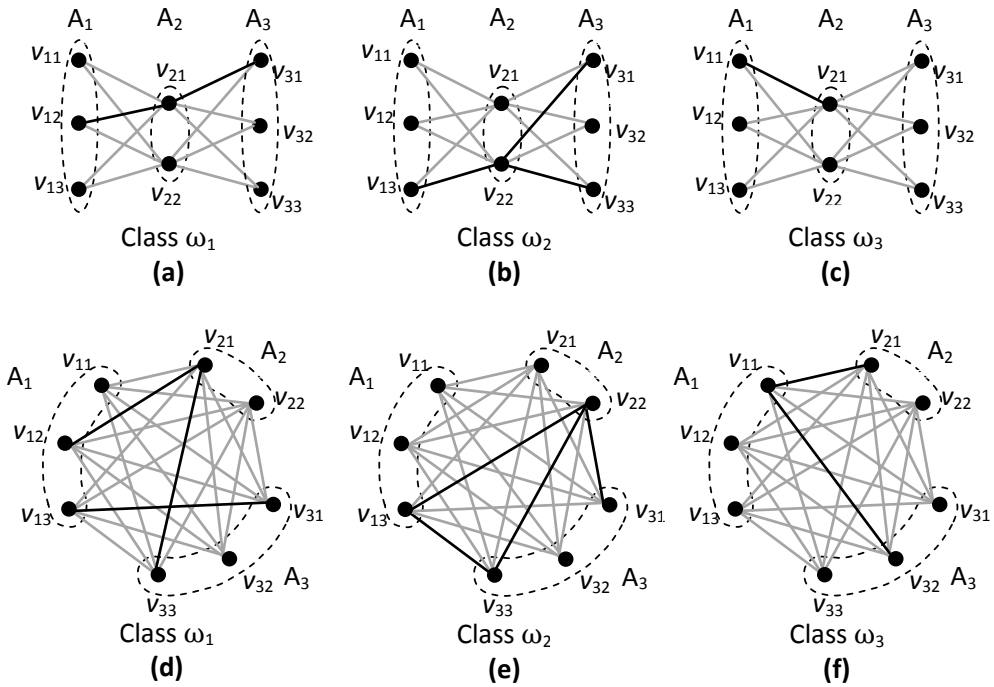
(a)



(b)

AbDG - motivation

- *Why building an AbDG?*

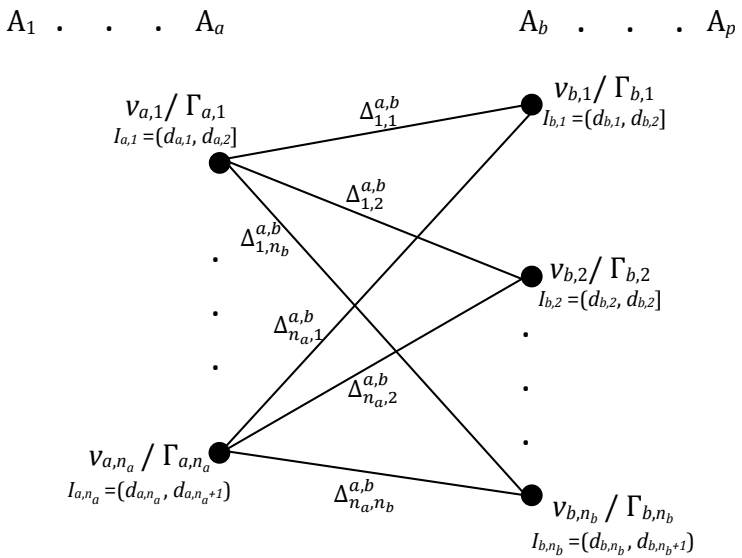


- Similar data instances produce similar subgraphs when projected onto the AbDG.
- A class can be represented by a particular set of such subgraphs.

Associating weights to the AbDG

A vector of weights with the size of the number of classes (M), is associated to every vertices and edges.

- Vertex $\mathbf{v_{a,i}}$ is associated to weight vector $\mathbf{\Gamma_{a,i}} = \langle \gamma_1, \ldots, \gamma_M \rangle$.
- Edge $(\mathbf{v_{a,k}}, \mathbf{v_{b,q}})$ is associated to weight vector $\mathbf{\Delta^{a,b}_{k,q}} = \langle \delta_1, \ldots, \delta_M \rangle$



Vertex weights

Vertex weight is the conditional probability of a given instance $\mathbf{x}_i$ to belong to class ω_j, given that $x_{i,a} \in I_{a,k}$

$$\Gamma_{a,k}(j) = P(\omega_j | I_{a,k}) \qquad (1)$$

Edge weights

Edge weight reflects the probability of a given data instance $\mathbf{x}_i$, whose attribute values $x_{i,a} \in I_{a,k}$ and $x_{i,b} \in I_{b,q}$ to belong to class ω_j

$$\Delta^{a,b}_{k,q}(j) = P(\omega_j | I_{a,k}, I_{b,q}) \qquad (2)$$

Determining the vertices weight

Considering a M-class problem

Each vertex $v_{a,i}$ $(1 \leq a \leq p$ and $1 \leq k \leq n_a)$ has a M-dimensional weight vector $\Gamma_{a,k} = \langle \gamma_1, \ldots, \gamma_j, \ldots, \gamma_M \rangle$ associated to it.

Let $I_{a,k}$ be the kth interval of attribute A_a, the weight γ_j is defined by:

$$\Gamma_{a,k}(j) = P(\omega_j | I_{a,k}) = \frac{P(I_{a,k}, \omega_j)}{P(I_{a,k})}$$

The joint probability $P(I_{a,k}, \omega_j)$, reflects the probability of an instance having class ω_j and value of attribute A_a lying in interval $I_{a,k}$.

$$P(I_{a,k}, \omega_j) = P(\omega_j) \frac{|\{\mathbf{x}_i | x_{i,a} \in I_{a,k} \wedge c_i = \omega_j\}|}{|\{\mathbf{x}_i | c_i = \omega_j\}|}$$

The normalizing term is the sum of the probabilities $P(I_{a,k}, \omega_j)$ for all classes.

$$P(I_{a,k}) = \sum_{j=1}^{M} P(I_{a,k}, \omega_j)$$

Determining the edges weights

Considering a M-class problem

Each edge has a weight vector $\Delta_{k,q}^{a,a+1} = \langle \delta_1, \ldots, \delta_M \rangle$, connecting the vertices $v_{a,k}$ and $v_{a+1,q}$, i.e. kth interval of attribute A_a and qth interval of attribute A_{a+1}

δ_i is defined as:

$$\Delta_{k,q}^{a,b}(j) = P(\omega_j | I_{a,k}, I_{b,q}) = \frac{P(I_{a,k}, I_{b,q}, \omega_j)}{P(I_{a,k}, I_{b,q})}$$

Since $P(I_{a,k}, I_{b,q}, \omega_j) = P(\omega_j)P(I_{a,k}, I_{b,q}|\omega_j)$, then define $P(I_{a,k}, I_{b,q}|\omega_j)$ as the ratio of instances belonging to class ω_i, whose values of attribute A_a lay within the kth interval and those of the attribute A_b lay within the qth interval, as in:

$$P(I_{a,k}, I_{b,q}|\omega_j) = \frac{|\{\mathbf{x}_i | c_i = \omega_j \wedge x_{i,a} \in I_{a,k} \wedge x_{i,b} \in I_{b,q}\}|}{|\{\mathbf{x}_i | c_i = \omega_j\}|}$$

An example of an AbDG induced from the Iris domain

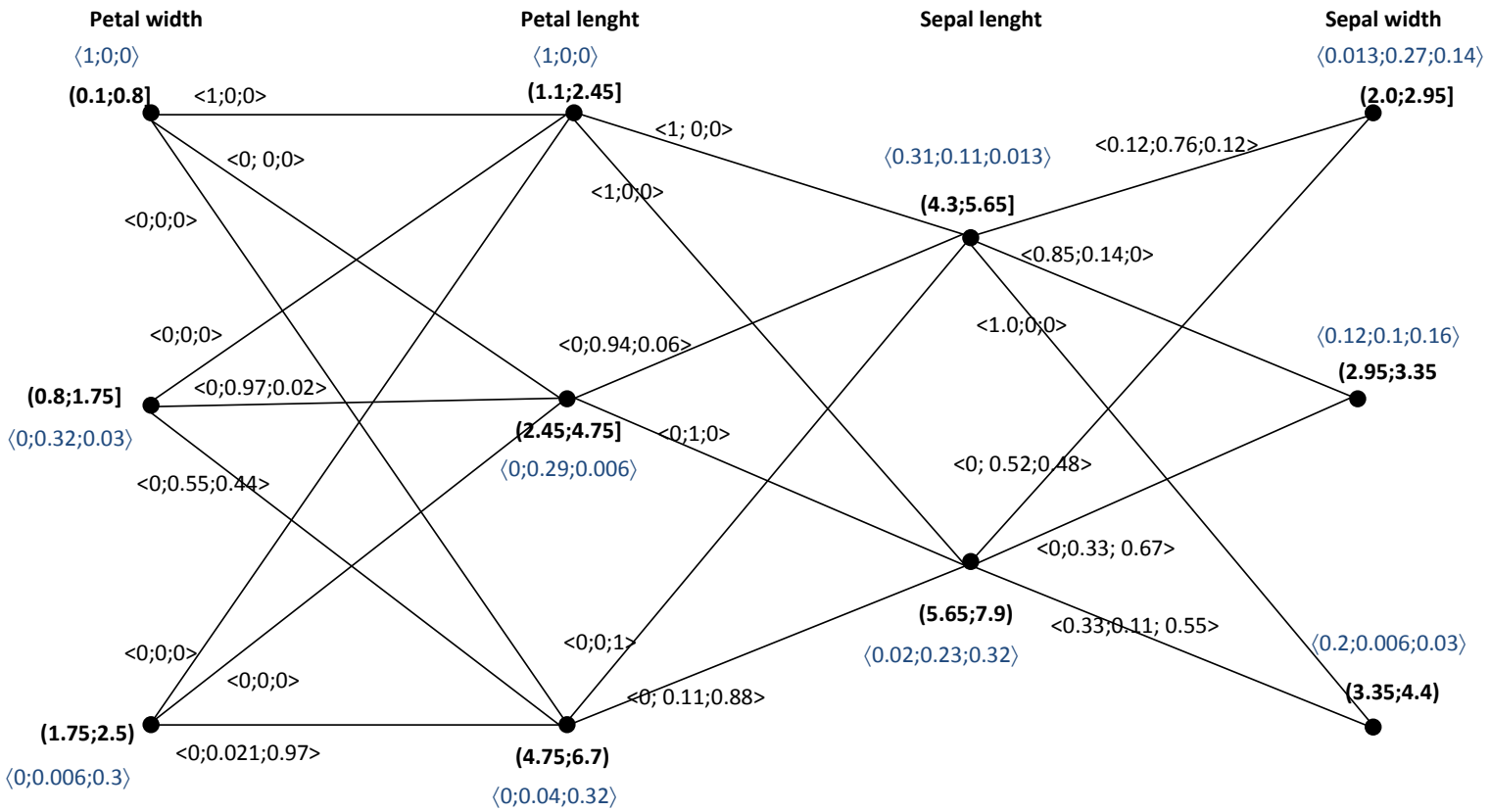
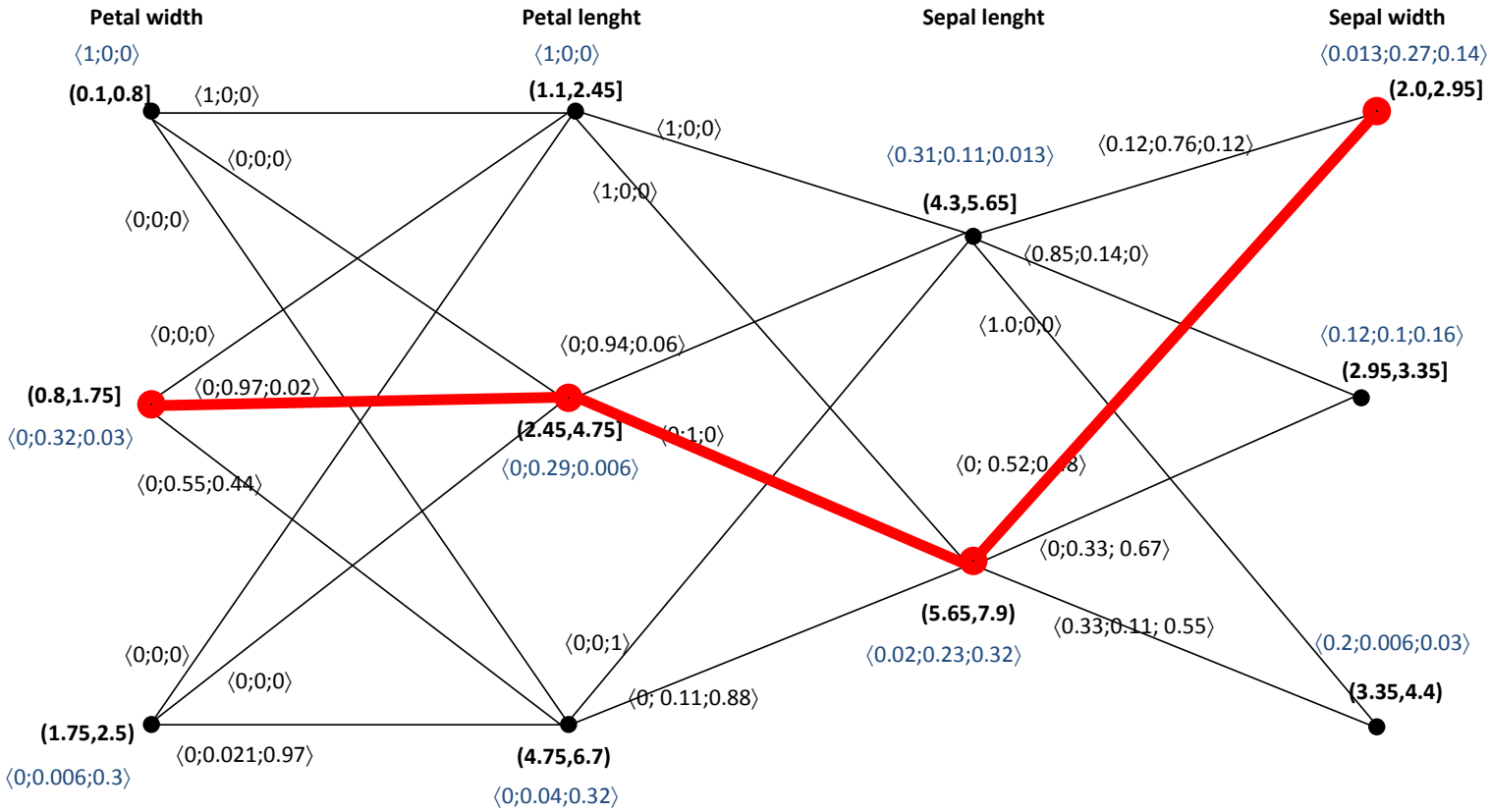


Figure: p-partite AbDG formed from the Iris domain.

Classification of stationary data

Using AbDG's as Classifiers

- Given an unlabeled pattern $\mathbf{y}$, the classification proceeds similar to a graph matching.
- For instance, consider classify pattern $\mathbf{y} = (1.2, 3.9, 5.8, 2.7)$ on the AbDG obtained from Iris.



Classification results

Results obtained with C4.5, M–I ID3, W-KNN, PNN, M-SVM and AbDG in 20 knowledge domains from the UCI-Repository. Each result is the classification accuracy rate averaged over repeated 10-fold cross-validation process followed by its standard deviation.

Domain	C4.5	M-I ID3	W-KNN	PNN	M-SVM	AbDG
Iris	95.5±4.9 (3)	93.9±5.6 (5)	92.1±4.2 (6)	95.0±4.6 (4)	**96.3±4.6** (1)	95.8±4.5 (2)
Wine	93.2± 5.90 (3)	92.9±5.6 (4)	66.9±4.8 (5)	62.2±10.4 (6)	97.7±2.9 (2)	**98.1±3.2** (1)
Balance	78.9±3.89 (4)	75.5±2.5 (5)	72.2±3.7 (6)	87.6±5.8 (3)	90.8±2.2 (2)	**91.7±1.3** (1)
Sonar	75.1±9.17 (4)	73.3±9.3 (5)	63.5±4.5 (6)	82.8±6.0 (2)	**84.6±5.7** (1)	81.7± 7.6 (3)
Credit	74.0±3.6 (4)	71.9±3.3 (5)	85.7±0.7 (2)	70.4±4.7 (6)	83.7±4.1 (3)	**89.0±3.9** (1)
Image	87.0±6.6 (3)	85.7±5.4 (4)	60.8±6.3 (6)	74.7±10.3 (5)	**90.0±4.7** (1)	87.3±6.3 (2)
Glass	68.6±7.6 (3)	61.7±9.3 (4)	54.8±5.6 (5)	32.3±5.8 (6)	70.3±8.2 (2)	**70.9±7.7** (1)
Pima	73.9±4.8 (4)	75.1±4.4 (2)	69.7±2.1 (5)	67.7±6.1 (6)	74.3±4.8 (3)	**76.5±4.6** (1)
WDBC	93.6±3.2 (3.5)	93.6±1.9 (3.5)	91.0±1.7 (5)	79.1±5.6 (6)	94.1±2.3 (2)	**95.0±2.9** (1)
WPBC	76.6±2.2 (4)	76.2±5.2 (5)	75.8±2.8 (6)	77.0±9.1 (3)	78.8±7.6 (2)	**79.2±5.2** (1)
Flags	63.4±8.8 (2)	62.5±9.7 (3)	55.1±6.3 (5)	40.0±8.2 (6)	60.9±13.1 (4)	**65.5±10.7** (1)
Waveform	71.6±6.9 (5)	64.4±6.1 (6)	75.3±3.5 (4)	76.0±6.9 (3)	80.0±6.3 (2)	**80.8±5.1** (1)
Heartspectf	72.3±14.3 (2)	71.2±14.5 (4)	68.7±8.2 (5)	57.5±24.4 (6)	71.7±12.4 (3)	**81.4±4.3** (1)
Soybean	92.1±3.1 (3)	91.7±3.1 (4)	72.7±3.2 (6)	91.3±2.6 (5)	**94.6±2.5** (1)	94.5±2.4 (2)
Segment	**96.5±2.1** (1)	91.5±1.9 (4)	86.1±1.2 (5)	78.7±3.7 (6)	93.6±1.1 (3)	94.2±1.3 (2)
Heartspect	**74.5±7.6** (1)	68.1±6.4 (6)	69.9±4.4 (5)	70.3±7.4 (4)	71.4±8.8 (2)	70.4±8.3 (3)
Blood	**78.6±3.5** (1)	76.2±1.5 (3.5)	72.4±2.4 (6)	76.2±5.2 (3.5)	75.7±1.4 (5)	77.8±3.3 (2)
Haberman	71.7±4.8 (6)	72.2±4.4 (5)	73.0±1.5 (3)	72.6±6.1 (4)	**75.8±8.0** (1)	75.3±5.8 (2)
Post Oper.	67.7±24.3 (3)	70.0±5.1 (2)	65.7±10.8 (5)	60.1±13.0 (6)	65.8±15.8 (4)	**73.3±7.5** (1)
Heart	83.3±9.1 (3)	75.1±8.8 (5)	76.4±4.0 (4)	55.5±11.6 (6)	**84.4±5.7** (1)	83.7±6.5 (2)
Average Rank	3.125	4.25	5.0	4.825	2.25	1.55

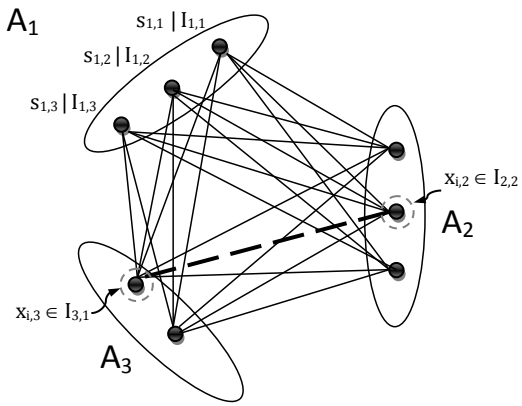
Imputation of missing data

Missing attribute values

- **Missing attribute values is a common problem in most of real applications**
- Facing this problem, two popular practices are employed:
  - **Discard** the patterns with missing values;
  - **Use an imputation method** to infer a plausible value.
- Usually, imputation methods are used in the pre-processing phase; prior to the learning algorithm;
  - Many imputation algorithm needs a reasonable sized set to induce the missing values;
- AbDG has mechanisms to handle missing values by itself.
- It can be used as an imputation method, to infer plausible values for missing data[6]

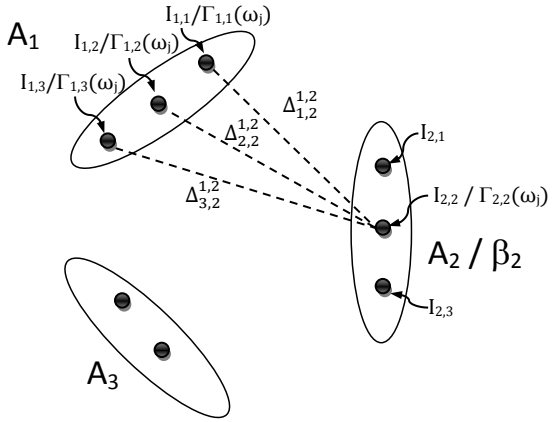
[6] Bertini JR, J. R.; Nicoletti, M. C. ; ZHAO, L. . An embedded imputation method via Attribute-based Decision Graphs. Expert Systems with Applications, v. 57, p. 159-177, 2016.

Performing imputation with the AbDG



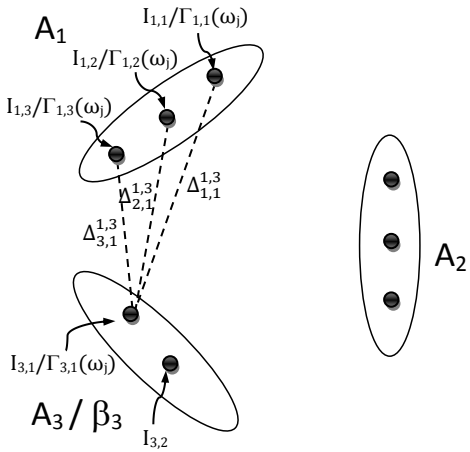
$\mathbf{x}_i = (?, x_{i,2}, x_{i,3}, c_i)$

(a)



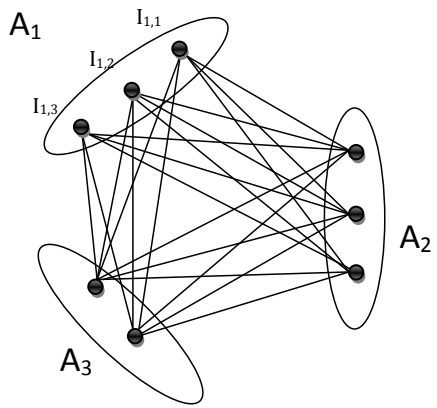
$s_{1,k} = s_{1,k} + \Gamma_{1,k}(\omega_j)\beta_2\Delta_{k,2}^{1,2}(\omega_j)\Gamma_{2,2}(\omega_j)$

(b)



$s_{1,k} = s_{1,k} + \Gamma_{1,k}(\omega_j)\beta_3\Delta_{k,1}^{1,3}(\omega_j)\Gamma_{3,1}(\omega_j)$

(c)

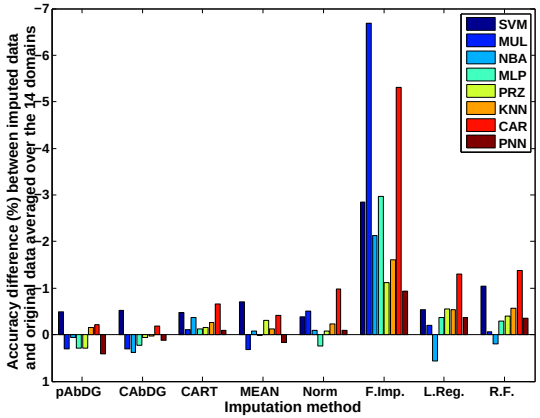


$[d_k, d_{k+1}] = I_{1,k} | \, max_{k=1,\ldots,3} s_{1,k}$

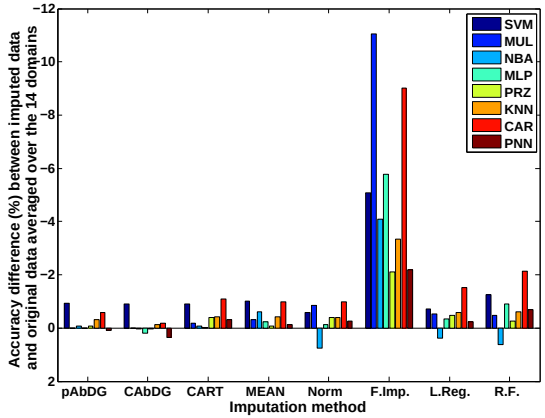
$\mathbf{x}_i = (x_{imp}, x_{i,2}, x_{i,3}, c_i) \qquad x_{imp} \in [d_k, d_{k+1}]$

(d)

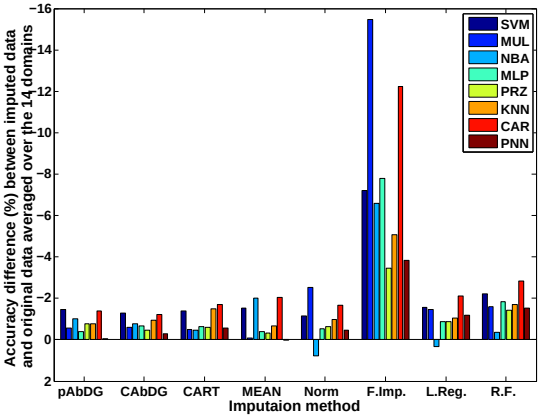
Imputation results



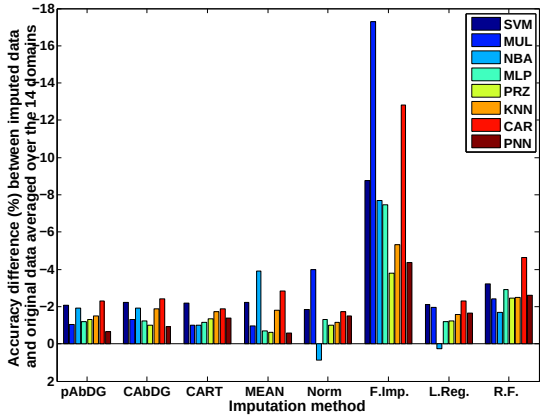
(a) 5%



(b) 10%



(c) 20%



(d) 30%

Imputation results

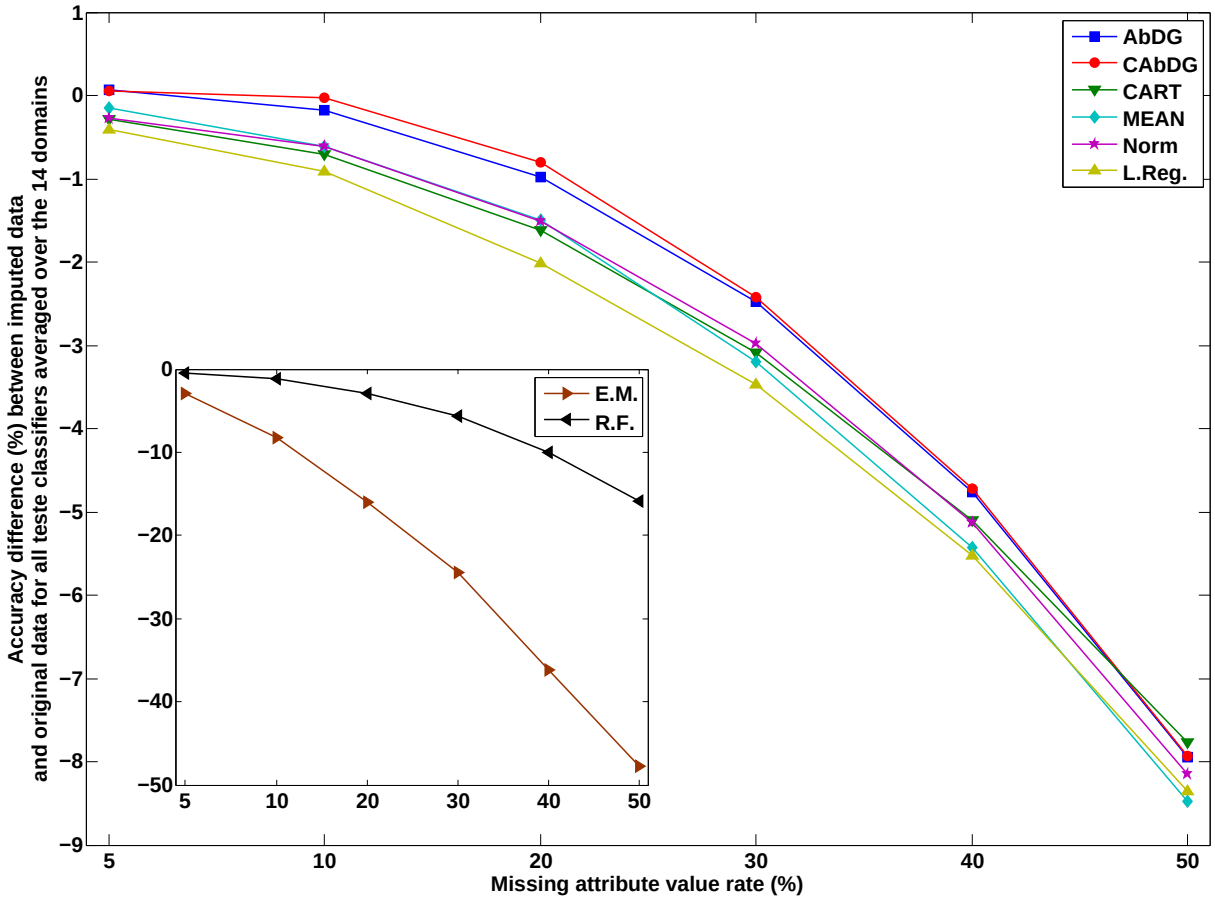


Figure: Differences in accuracy between imputed data and original data averaged over all obtained results, for each missing attribute rate and imputation method.

Enhancing data quality for classification tasks

Enhancing data quality with the AbDG

- **The accuracy performance of an automatic classification system strongly relies on the quality of the data set used to train it!**
  - Pre-processing tasks referred to as data cleansing, consist of searching for values that are missing, corrupted, inconsistent or inaccurate, and providing mechanisms for correcting them;
  - Attribute noise detection can be a particularly difficult task, once that noisy data may look like regular data.

- **Idea:** using the AbDG, check every attribute value for inconsistency, through a procedure inspired in the AbDG imputation process.[7]
  - The cleansing method systematically infers an interval of values for each attribute value and verifies if the real value belongs to the inferred interval.

[7] Bertini Jr., J.R. A methodology for enhancing data quality for classification purposes using attribute-based decision graphs. In: 2017 IEEE Latin American Conference on Computational Intelligence (LACCI), 2017.

Enhancing data quality with the AbDG

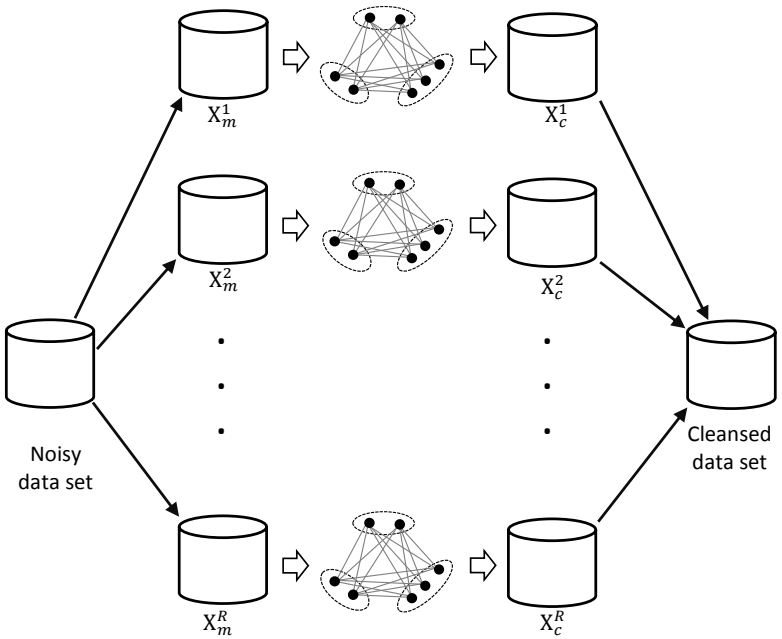


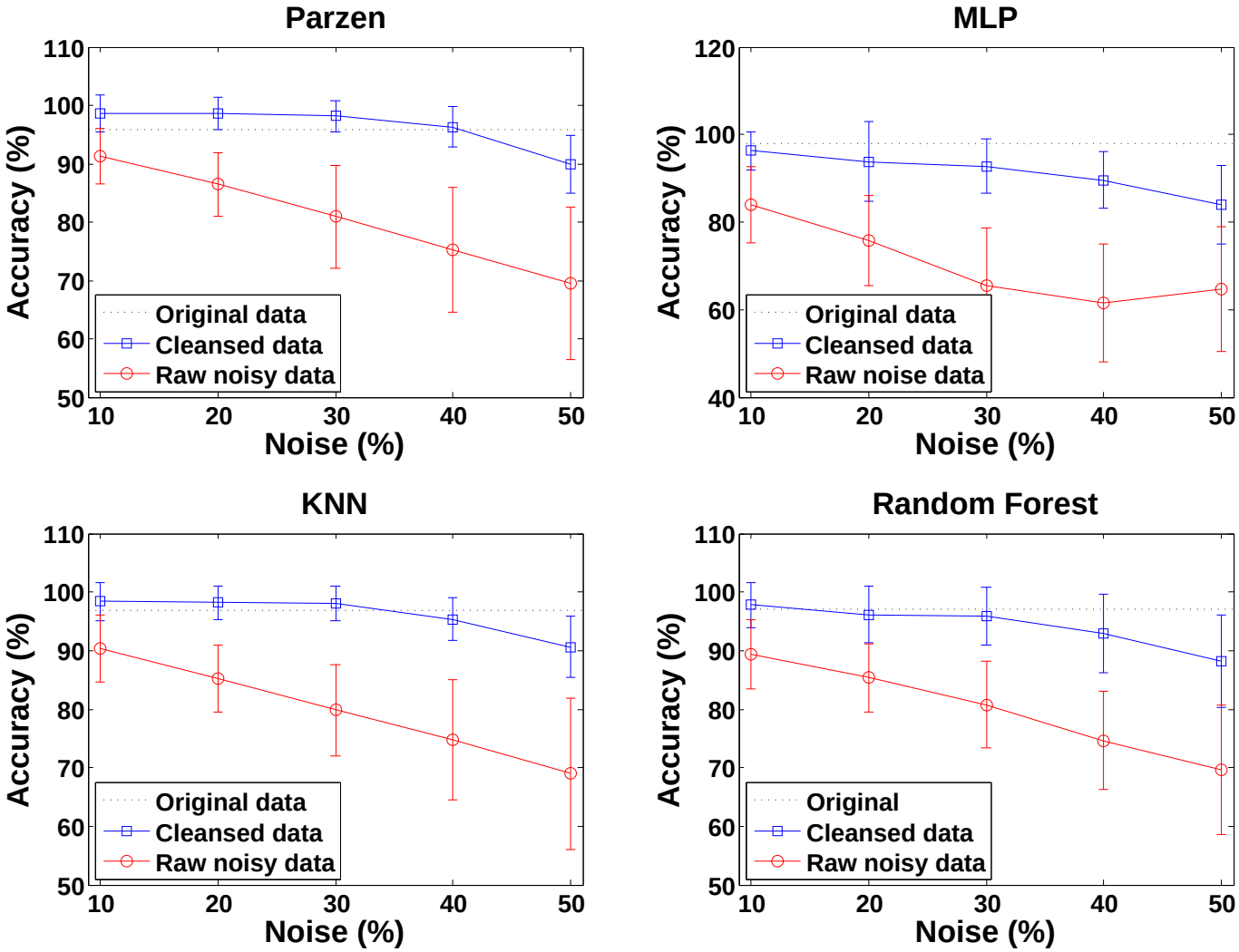
Figure: Cleansing procedure conducted through a CAbDG. Run for T times and join them using the modal interval.

Cleansing procedure based on the AbDG

- Given a noisy data set X.
- Let D^r be a subset of attribute values, where $X = D^1 \cup \cdots \cup D^R$ and $D^1 \cap \cdots \cap D^R = \emptyset$, for $r = 1 \ldots R$.
- For each D^r build a CAbDG with the set $X_m^r = X - D^r$, referred to G_r.
- Building X_c^r: for each value in D^r, use G_r for estimating an interval of values into which it should rely.
  - If it relies within the inferred interval, keep it!.
  - Otherwise, draw a random value within the interval.

Data quality results

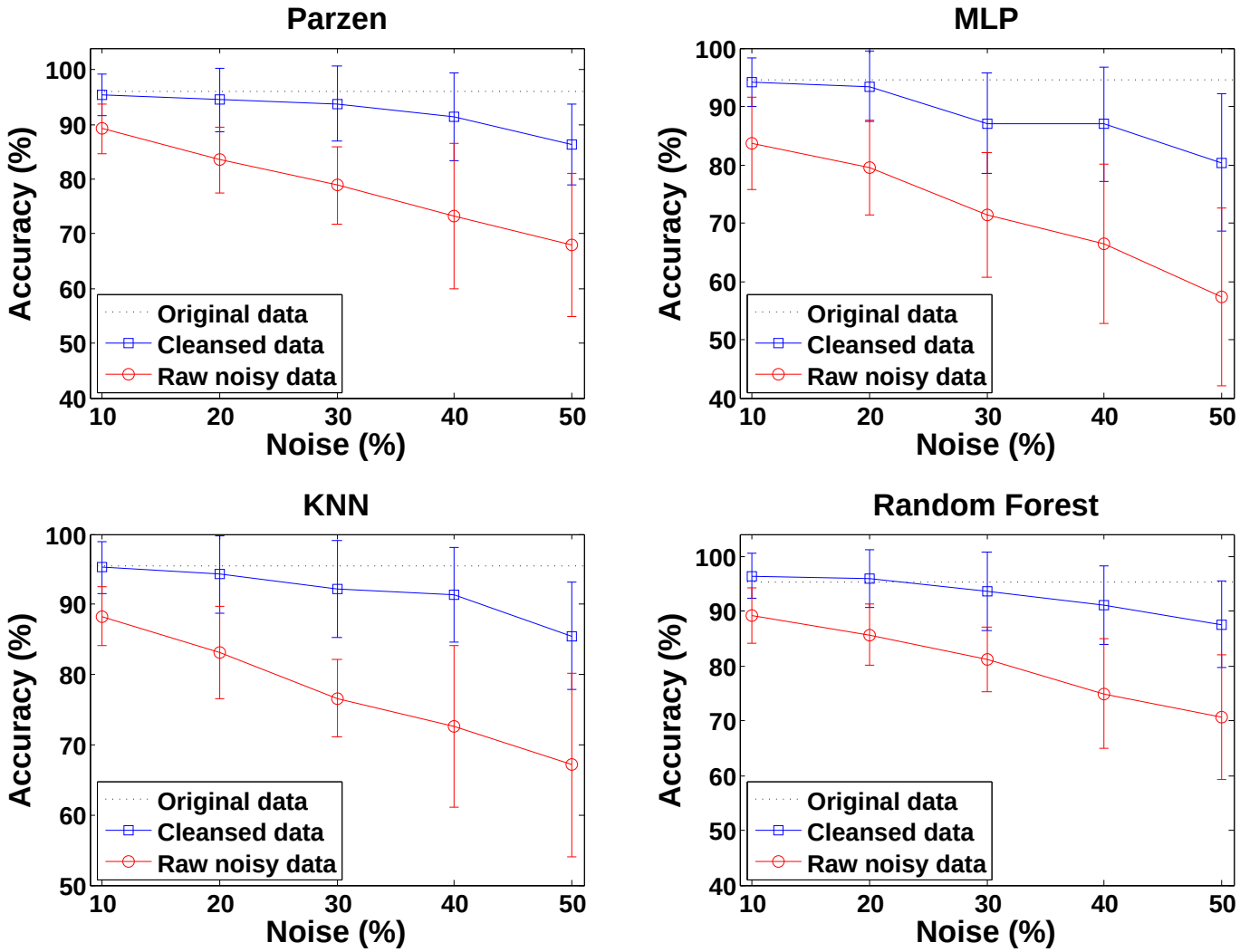
Wine domain with the following level of noise: 10%, 20%, 30%, 40%, 50% .



Classification accuracy results for the algorithms Parzen, MLP, KNN and Random Forest when trained with the cleansed data

Data quality results

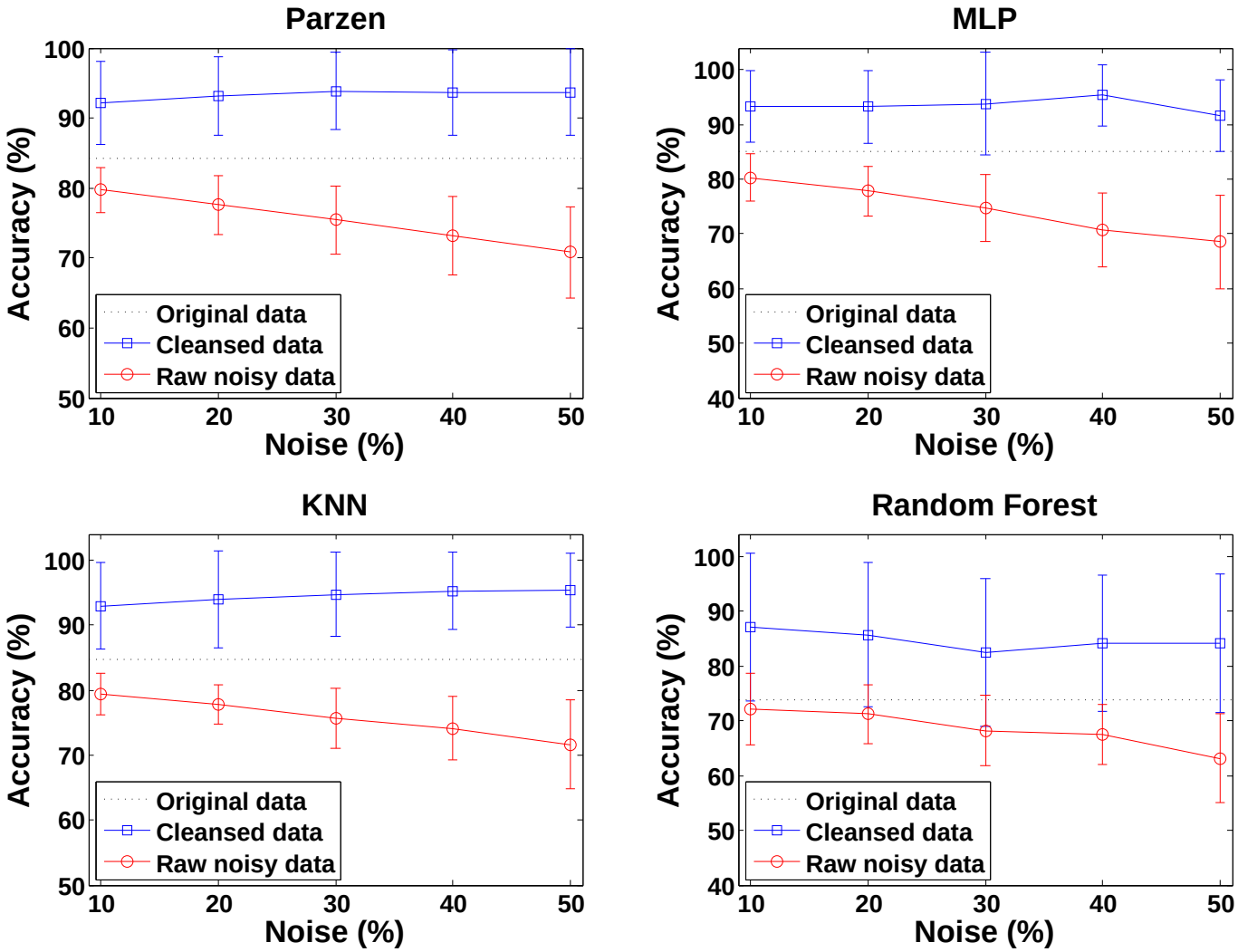
Iris domain with the following level of noise: 10%, 20%, 30%, 40%, 50% .



Classification accuracy results for the algorithms Parzen, MLP, KNN and Random Forest when trained with the cleansed data

Data quality results

Credit domain with the following level of noise: 10%, 20%, 30%, 40%, 50% .



Classification accuracy results for the algorithms Parzen, MLP, KNN and Random Forest when trained with the cleansed data

Conclusions and future directions

Conclusions and future directions

- The K-associated graphs and Attribute-based Decision Graphs are alternative graph models that have been successfully applied to machine learning tasks, as:
  - Supervised and semi-supervised classification.
  - Learning from data stream.
  - Imputation of missing data.
  - Enhancement of data quality for classificaion.

- Future directions.
  - Rule extraction from AbDG for white-box classification in data stream.
  - Active learning, using the K-associated approach, to cope with data streams with few labeled data.
  - Applying AbDG and K-associated graphs to pattern recognition (image/video).
  - Extending initial results.

Thanks

Q & A
Thanks for your attention!