Monotonic Ordinal Classification

A Path to Fairness in Machine Learning Prediction

Salvador García

DaSCI: Andalusian Research Institute in Data Science and Computational Intelligence
University of Granada, Spain.

Preliminaries

A Comprehensive Taxonomic Overview

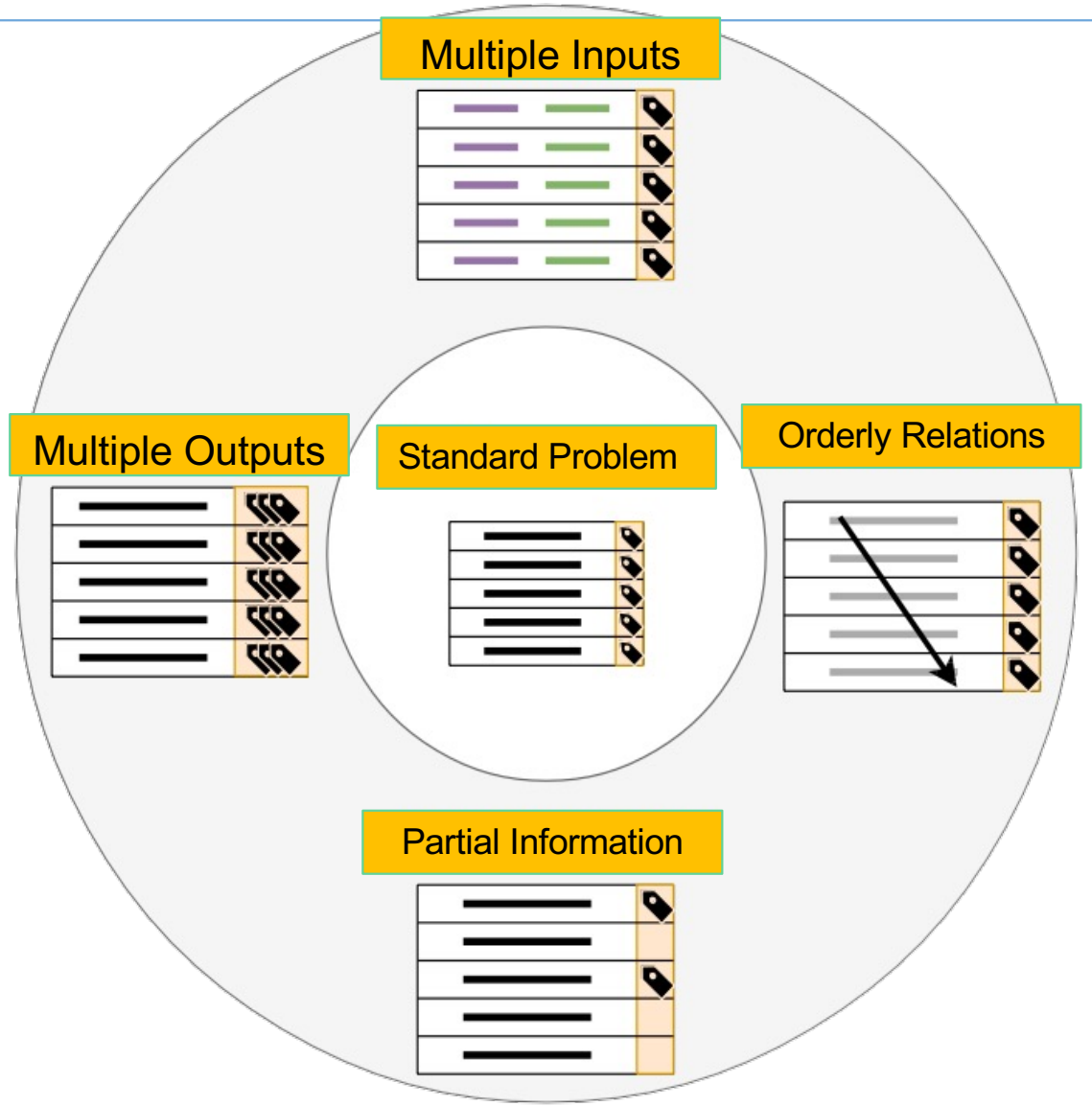
Some MOC Proposals

MOC and Fairness in ML

New Paradigms in MOC

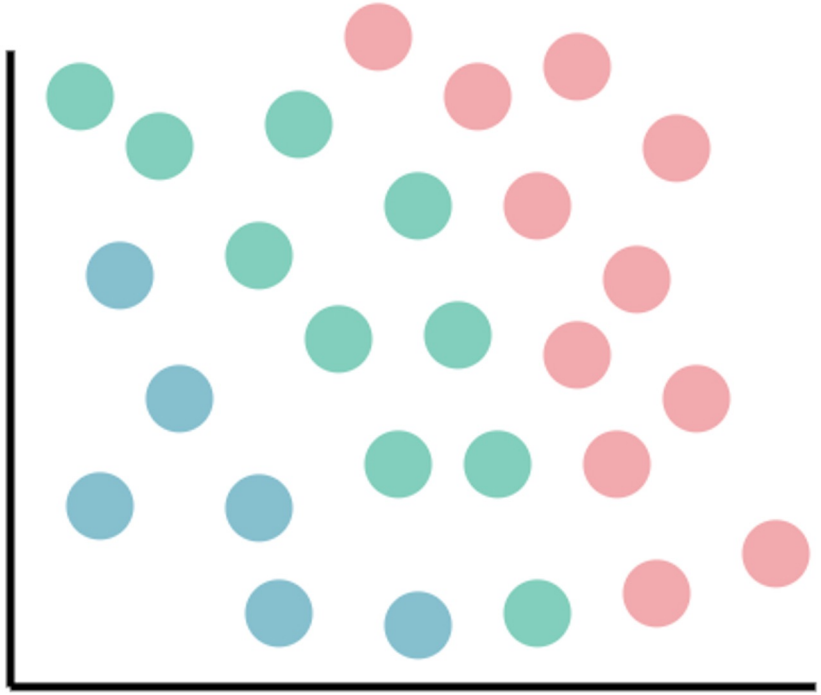
Conclusions & Future Work

They arise from variations in the input and output structures that do not fit the standard problem.



Preliminaries

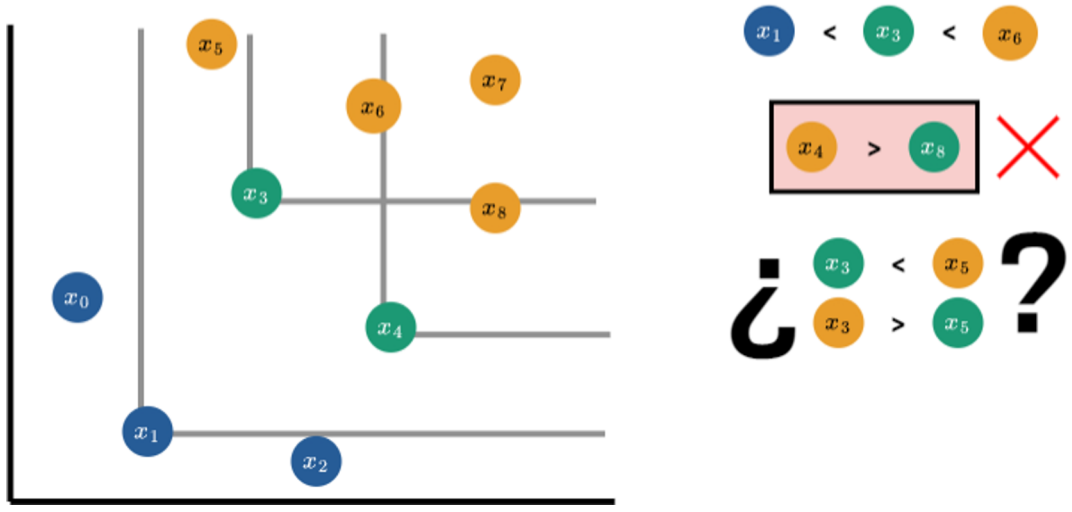
Ordinal Classification or Regression



- There is an order relationship between classes.
- Our goal is to minimize the number of misclassifications regarding the class order.
- The costs of misclassifications are different for every class.

Preliminaries

Monotonicity in Ordinal Classification



- There is an order relationship between classes.
- There exists an order relationship between instances
- If two instances are comparable, then their labels have to be assigned according to the order of this comparison
- There are partial order variations, with monotonic and non-monotonic features.

Now, we formally define a classification dataset with ordinal labels and monotonicity constraints. Let assume that patterns are described using a total of f input variables with ordered domains, $\mathbf{x}_i \subseteq \mathbb{R}^f$, and a class label, y_i, from a finite set of C ordered labels, $y_i \in \mathcal{Y} = \{1, ..., C\}$. In this way, the data set D consists of n samples or instances $D = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$. As previously discussed, a *dominance relation*, $\succeq$, is defined as follows:

$$\mathbf{x} \succeq \mathbf{x}' \Leftrightarrow x^s \geq x^{s'} \forall s \text{ with a monotonicity constraint,} \tag{1}$$

where x^s and $x^{s'}$ are the s-th coordinates of patterns $\mathbf{x}$ and $\mathbf{x}'$, respectively. In other words, $\mathbf{x}$ dominates $\mathbf{x}$' if each coordinate of $\mathbf{x}$ is not smaller than the respective coordinate of $\mathbf{x}'$.

Definition of Monotony

Partial Orders

- In order to apply monotonicity to vectors (instances), the natural (total) order must be extended to allow for vector orders in $\mathbb{R}^p$

- These relationships are called partial orders

$$\mathbf{x} \preceq \mathbf{x}' \iff \forall i = 1..p, \;\; x_i \leq x'_i$$

- Full monotonic function:

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow f(\mathbf{x}) \leq f(\mathbf{x}'), \qquad \forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^p$$

- Properties:

1. *Reflexivity:* $\mathbf{a} \preceq \mathbf{a}$
2. *Transitivity:* $\mathbf{a} \preceq \mathbf{b} \wedge \mathbf{b} \preceq \mathbf{c} \Rightarrow \mathbf{a} \preceq \mathbf{c}$
3. *Antisymmetry:* $\mathbf{a} \preceq \mathbf{b} \wedge \mathbf{b} \preceq \mathbf{a} \Rightarrow \mathbf{a} = \mathbf{b}$

Definition of Monotony

Samples $\mathbf{x}$ and $\mathbf{x}'$ in space D are *comparable* if either $\mathbf{x}' \succeq \mathbf{x}'$ or $\mathbf{x}' \succeq \mathbf{x}$. Both $\mathbf{x}$ and $\mathbf{x}'$ are *incomparable* otherwise. Two examples $\mathbf{x}$ and $\mathbf{x}'$ are *identical* if $x^j = x^{j'}, \forall j \in \{1, \ldots, f\}$, and they are *non-identical* if $\exists j$ for which $x^j \neq x^{j'}$.

A pair of comparable examples $(\mathbf{x}, y)$ and $(\mathbf{x}', y')$ is said to be monotone if [1]:

$$\mathbf{x} \succeq \mathbf{x}' \wedge \mathbf{x} \neq \mathbf{x}' \wedge y \geq y', \tag{2}$$

or

$$\mathbf{x} = \mathbf{x}' \wedge y = y'. \tag{3}$$

A data set D with n examples is monotone if all possible pairs of examples are either monotone or incomparable. It is worth mentioning that the previous notation was expressed for direct monotonicity constraints, but it could be changed to consider inverse ones.

Definition of Monotony

Partially Monotonic Functions

- There can be monotonic and non-monotonic attributes.
- The input space can be divided into X (monotonic attributes) and Z (non-monotonic attributes).

- The partial order is now defined as:

$$(\mathbf{x}, \mathbf{z}) \preceq_{\mathcal{X}} (\mathbf{x}', \mathbf{z}') \quad \Leftrightarrow \quad \mathbf{x} \preceq \mathbf{x}' \wedge \mathbf{z} = \mathbf{z}'$$

- The monotonic function is defined as:

$$\boldsymbol{x} \preceq \boldsymbol{x}' \Rightarrow f(\boldsymbol{x}, \boldsymbol{z}) \leq f(\boldsymbol{x}', \boldsymbol{z}), \qquad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}, \boldsymbol{z} \in \mathcal{Z}$$

Preliminaries

Pioneers algorithms: Monotonic Induction of Decision Trees

- **DEFINITION 1**: Let $X = (x_1, x_2, \ldots, x_n)$ *and* $Y = (y_1, y_2, \ldots, y_n)$ two instances of the same problem, with n attributes. All attribute values are ordinal or numeric. An order between X and Y is defined as:

$$
\begin{array}{lll}
X = Y & \text{if } x_i = y_i & \forall i = 1, 2, \ldots, n \\
X > Y & \text{if } x_i > y_i & \forall i = 1, 2, \ldots, n \\
X \geq Y & \text{if } x_i > y_i \quad \text{OR} \quad x_i = y_i & \forall = 1, 2, \ldots, n \\
X < Y & \text{if } x_i < y_i & \forall i = 1, 2, \ldots, n \\
X \leq Y & \text{if } x_i < y_i \quad \text{OR} \quad x_i = y_i & \forall i = 1, 2, \ldots, n
\end{array}
$$

- **DEFINITION 2**: Let $X = (X, C_x)$ *and* $Y = (Y, C_y)$ represent two class-attribute pairs. The classes of X and Y are denoted by C_x and C_y respectively. (X, C_x) *and* (Y, C_y) are non-monotonic with respect to each other if:

$$
\begin{array}{ll}
X \leq Y \wedge C_x > C_y & \text{OR} \\
X \geq Y \wedge C_x < C_y & \text{OR} \\
X = Y \wedge C_x \neq C_y &
\end{array}
$$

- **DEFINITION 3**: Two class-attribute pairs (X, C_x) *and* (Y, C_y) are monotonic with respect to each other if either of the conditions of the above definition is not satisfied.

Preliminaries

Pioneers algorithms: Monotonic Induction of Decision Trees

- **DEFINITION 4**: Let *(P, C_p) and (Q, C_q)* two attribute-test/response-node paths in the same decision tree, where *P* and *Q* are attribute-test and C_p *and* C_q are answer-nodes. Both paths are monotonic with respect to each other if either of the conditions of definition 2 is not satisfied.

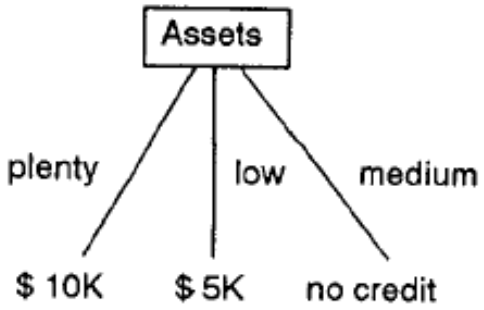
- **DEFINITION 5**: A decision tree is monotonic if all attribute-test/response-node pairs are monotonic with respect to each other.

Monotonic data set: Comparable examples are monotonic (#1 vs. #2, #1 vs. #3) or non-comparable (#2 vs. #3).

Example	Income	Assets	Credit history	Class
#1	high	plenty	good	$ 10K
#2	high	low	bad	$ 5K
#3	low	medium	bad	no credit

If we apply ID3:

- E(income) = 0.667 (2/3*1 + 1/3*0)
- E(assets) = 0



Building accurate monotonic trees

- **DEFINITION 6:** A non-monotonicity index is the ratio between the actual number of pairs of branches in the decision tree, and the maximum number of branches that could be non-monotonic with respect to each other in the same tree.
- To find this index on a k-branch tree, we construct a matrix *M*, *k x k* non-monotonic and symmetric, where the element m_{ij} is 1 if the *i-th branch* is non-monotonic with respect to the branch *j-th*, and 0 vice-versa. The sum of the *M*'s entries is denoted by *W.*

$$W = \sum_{i=1}^{k} \sum_{j=1}^{k} m_{ij}$$

- At most, $(k^2 - k)$ entries of M can be labelled as non-monotonic. The non-monotonicity index of a decision tree with attributes test $a_1, a_2, \ldots, a_v$, is defined as:

$$I_{a_1,a_2,\ldots,a_v} = \frac{W_{a_1,a_2,\ldots,a_v}}{k^2_{a_1,a_2,\ldots,a_v} - k_{a_1,a_2,\ldots,a_v}}$$

Preliminaries

Pioneers algorithms: Monotonic Induction of Decision Trees

- **DEFINITION 7**: The order ambiguity score of a decision tree is defined in terms of the non-monotonicity index.

$$A_{a_1,a_2,\ldots,a_v} = \begin{cases} 0 & \text{if } I_{a_1,a_2,\ldots,a_v} = 0 \\ -(\log_2 I_{a_1,a_2,\ldots,a_v})^{-1} & \text{otherwise} \end{cases}$$

- **DEFINITION 8**: The total ambiguity score is the sum of the entropy, used by ID3 or C4.5, and the order ambiguity score.

$$T_{a_1,a_2,\ldots,a_v} = E_{a_1,a_2,\ldots,a_v} + A_{a_1,a_2,\ldots,a_v}$$

- An effective way to express trade-offs between entropy and monotonicity can be achieved by introducing an additional parameter to the total ambiguity score.

$$T_{a_1,a_2,\ldots,a_v} = E_{a_1,a_2,\ldots,a_v} + R A_{a_1,a_2,\ldots,a_v}$$

- The parameter R expresses the relative importance of monotonicity relative to the inductive hit on a given problem. This parameter could be adjusted by several iterations of the algorithm on the same data.

Preliminaries

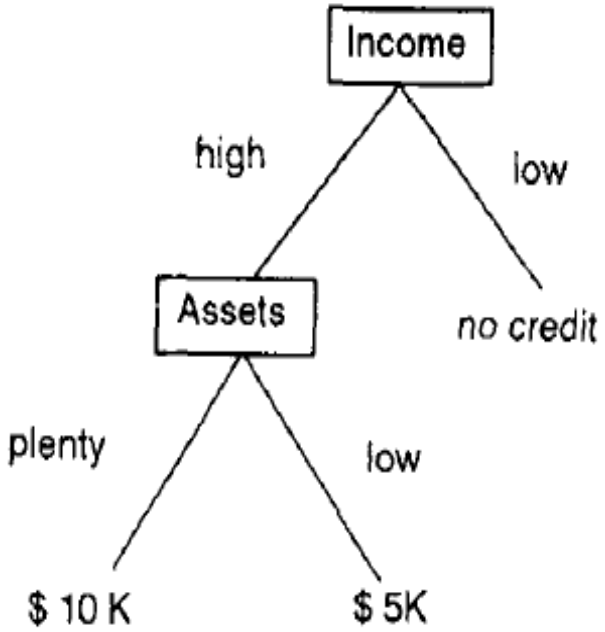
Pioneers algorithms: Monotonic Induction of Decision Trees

- To illustrate how this works, we apply ID3 to the above data. We use R = 2:

$$E_{\text{income}} = 0.667 \text{ bits}$$
$$I_{\text{income}} = A_{\text{income}} = 0$$
$$T_{\text{income}} = 0.667\ (0.667 + 2 * 0)$$
$$E_{\text{assets}} = 0 \text{ bits}$$
$$I_{\text{assets}} = 0.333\ (2/(3^2 - 3))$$
$$A_{\text{assets}} = 0.630\ (-\log_2 0.333)^{-1}$$
$$T_{\text{assets}} = 1.260\ (0 + 2 * 0.630)$$
$$E_{\text{credit history}} = 0.667 \text{ bits}$$
$$I_{\text{credit history}} = A_{\text{credit history}} = 0$$
$$T_{\text{credit history}} = 0.667.$$

- The total ambiguity scores for income and credit history are the lowest, and we assume that the income attribute is selected in the first partition. The total ambiguity score of income + assets is tested:

$$E_{\text{income+assets}} = 0 \text{ bits}$$
$$I_{\text{income+assets}} = A_{\text{income+assets}} = 0$$
$$T_{\text{income+assets}} = 0.$$

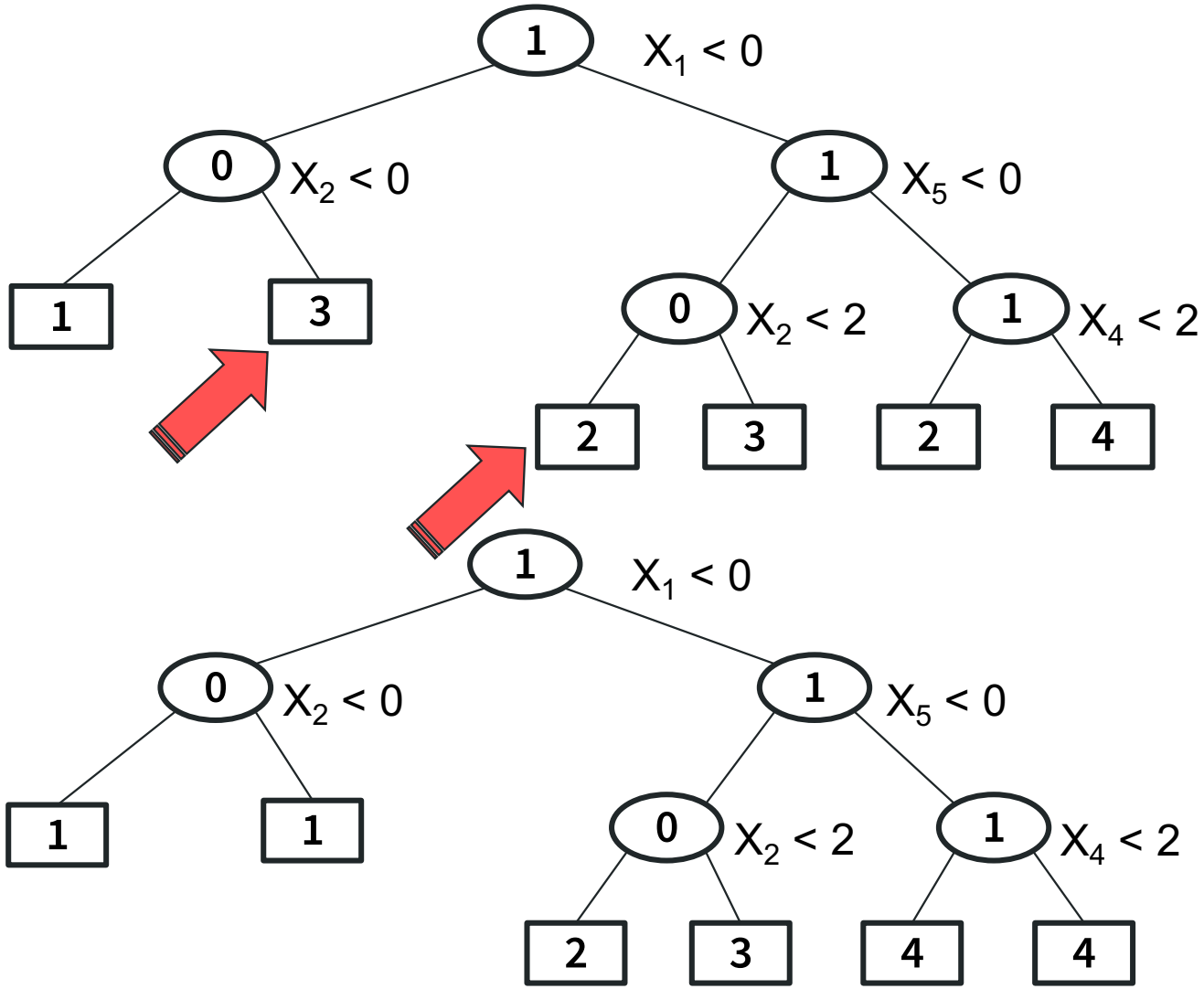


- Now, the tree obtained is monotonic.

NON-MONOTONE DECISION TREE

(-1,0,0,0,-2) < (1,1,1,1,-1)

MONOTONE DECISION TREE



Monotonic Data Set: Auto-MPG

Attribute	Type	Sign
mpg	continuous	target
cylinders	multi-valued discrete	–
displacement	continuous	–
horsepower	continuous	–
weight	continuous	–
acceleration	continuous	+
model year	multi-valued discrete	+
origin	multi-valued discrete	+

The relationship of monotony between each predictor variable and the target variable has a sign (positive = direct relationship, negative = inverse relationship)

Mean Zero One Error

$$MZE = 1 - Acc = \frac{1}{N}\sum_{i=1}^{N}[\![y_i^* \neq y_i]\!]$$

Mean Absolute Error

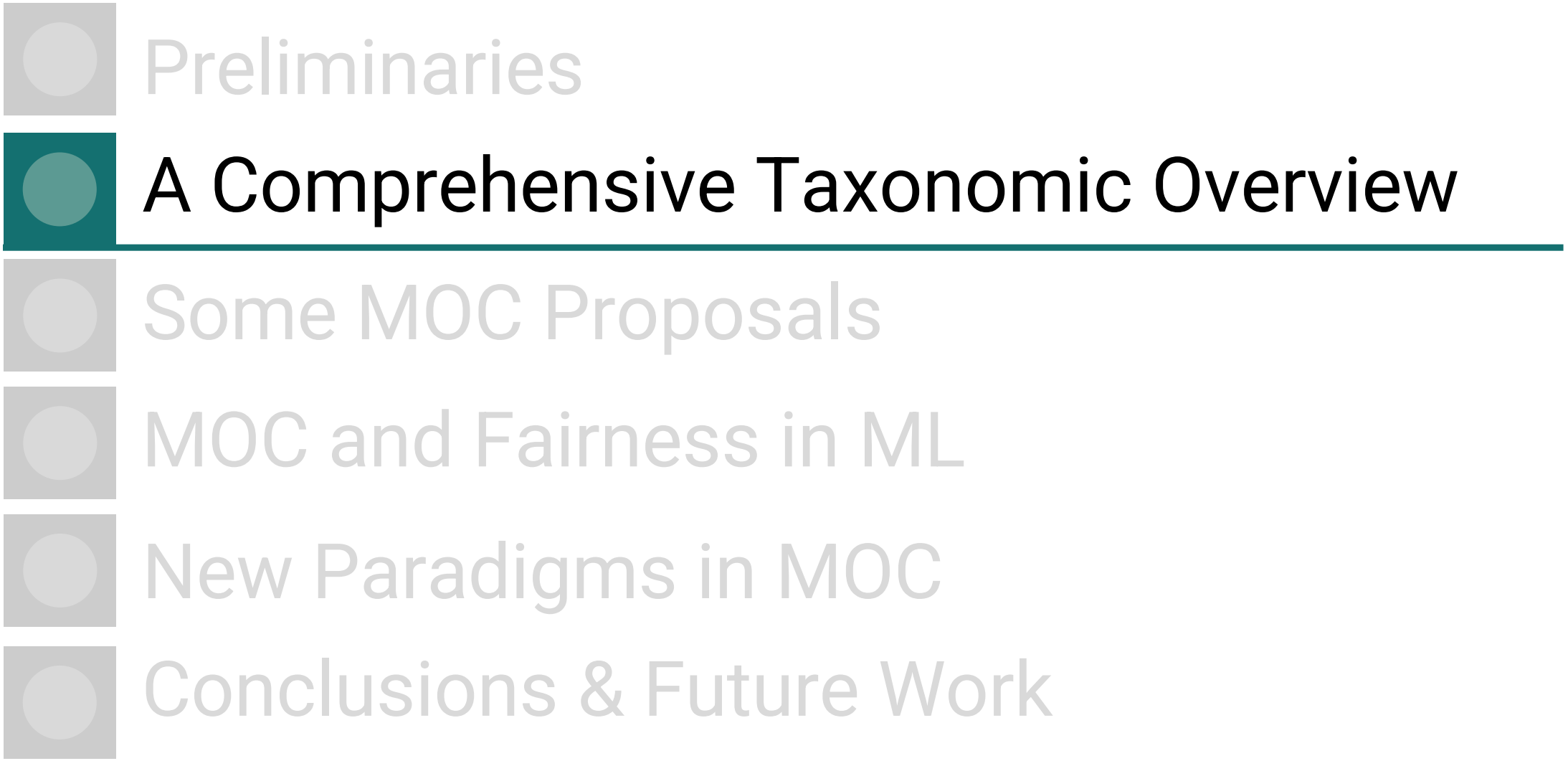
$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\mathcal{O}(y_i) - \mathcal{O}(y_i^*)|$$

Non-Monotone Indexes

$$NMI1 = \frac{1}{N^2 - N}\sum_{i=1}\sum_{j=1}\mathbf{1}[(\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j)]$$

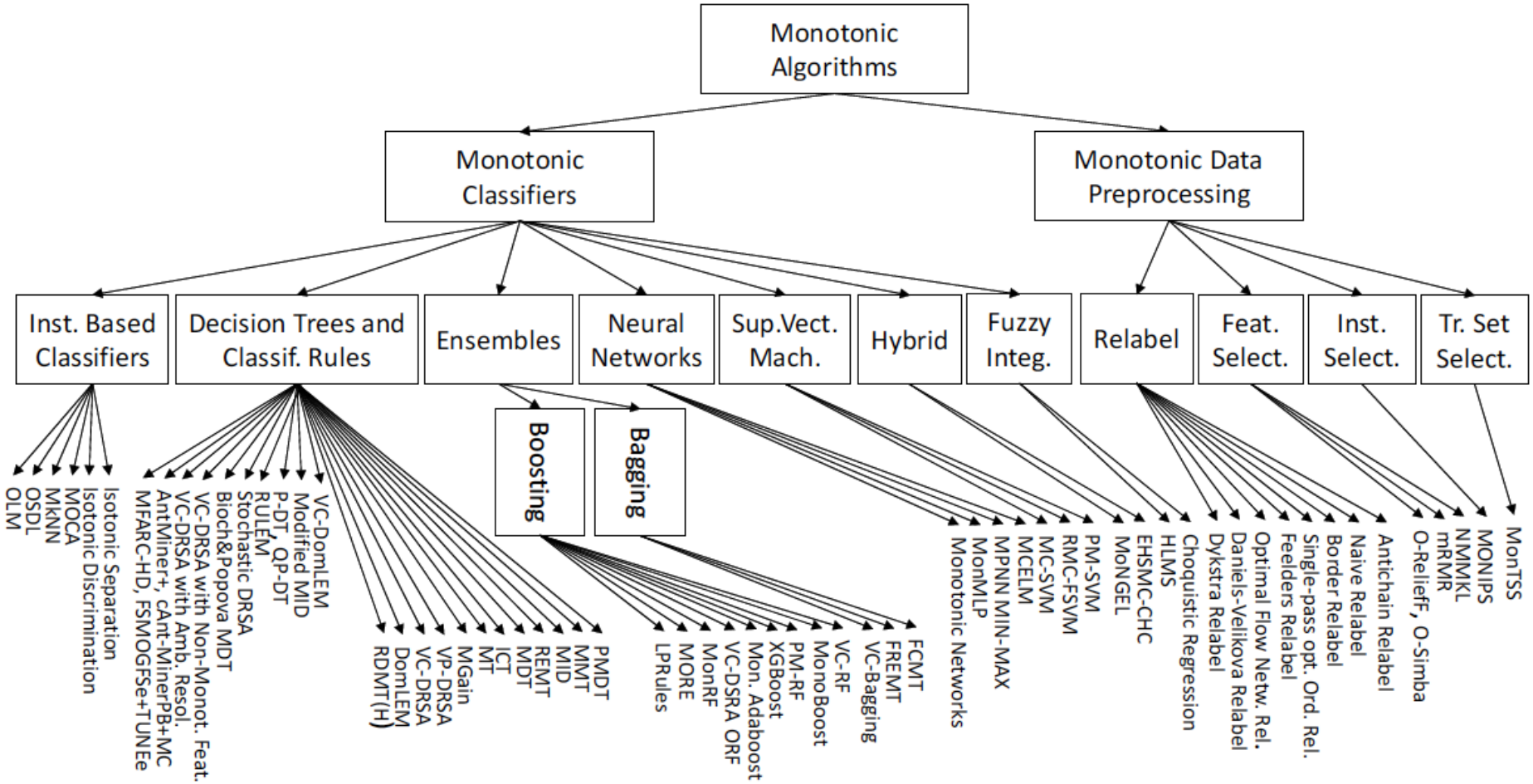
$$NMI2 = \frac{1}{NumComparablePairs}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{1}[(\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j]$$

$$NMI3 = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}[\exists \mathbf{x}_j \ \ s.t. \ (\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j)]$$


Preliminaries
A Comprehensive Taxonomic Overview
Some MOC Proposals
MOC and Fairness in ML
New Paradigms in MOC
Conclusions & Future Work

A Comprehensive Taxonomic Overview

Taxonomic Tree



A Comprehensive Taxonomic Overview

Statistics and curious facts

Table 4
Number of times each metric is used in monotonic classification literature.

Metric	# of times used
Accuracy	24
MAE	21
Error rate	5
κ coefficient	4
MSE	4
Recall	3
F-measure	3
PPV	2
MMAE	2
AUC	2
NPV	1
MAcc	1
NMI	8
MCC	2
γ_1	1
γ_2	1
FOM	1
NMI2	0

Table 5
Summary of the most used data sets used in the monotonic classifiers literature.

Data set	Ex.	Atts.	Num.	Nom.	Cl.	Source	NMI	# of times used
AutoMPG	392	7	7	0	10	[105]	0.023	17
BostonHousing	506	12	10	2	4	[106]	0.001	15
Car	1728	6	0	6	4	[105]	0.000	22
ERA	1000	4	4	0	9	[69]	0.016	15
ESL	488	4	4	0	9	[69]	0.004	18
LEV	1000	4	4	0	5	[69]	0.006	15
MachineCPU	209	6	6	0	4	[105]	0.001	19
Pima	768	8	8	0	2	[105]	0.015	16
SWD	1000	10	10	0	4	[69]	0.009	16

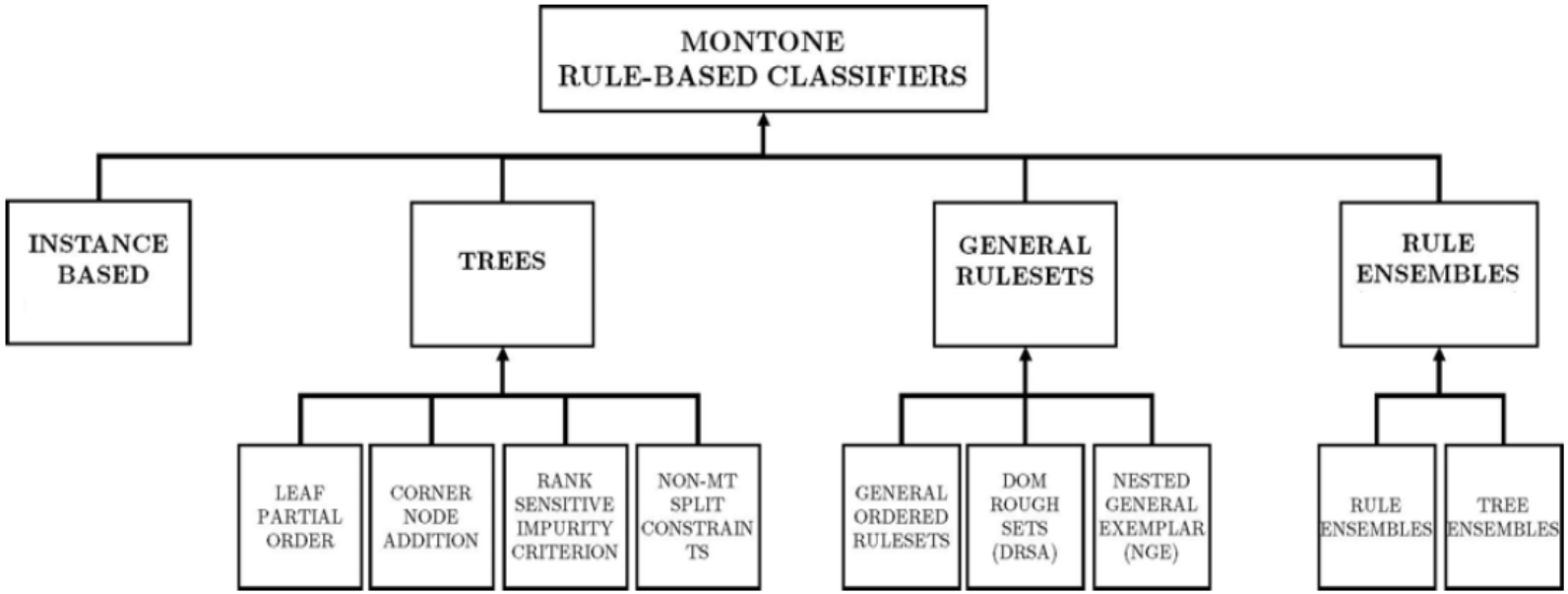
Monotonic classification: An overview on algorithms, performance measures and data sets
JR Cano, PA Gutiérrez, B Krawczyk, M Woźniak, S García
Neurocomputing 341, 168-182 (2019)

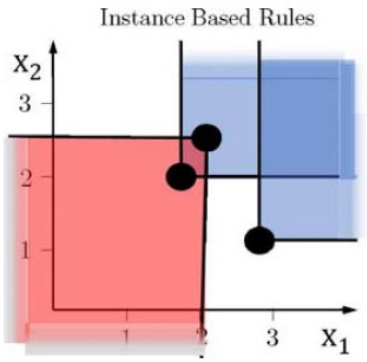
Table 3
Metrics considered in the reviewed monotonic classification methods.

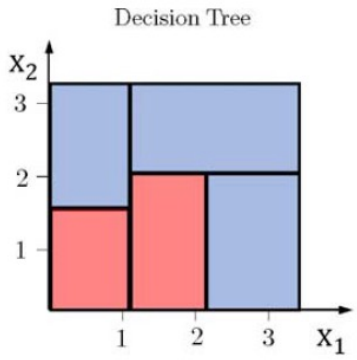
Abbr. name	Predictive assessment metrics	Monotonicity fulfillment metrics
OLM	MSE	None
MID	MSE, MAE	NMI
HLMS	Accuracy	None
Monotonic networks	Error rate	None
P-DT, QP-DT	Error rate	None
Isotonic discrimination	None	None
MT	Accuracy	None
VC-DRSA	None	None
DomLEM	None	None
Bioch&Popova MDT	None	None
Modified MID	Error rate	NMI
MDT	Accuracy	γ_1, γ_2
Isotonic separation	None	None
MonMLP	None	None
VC-DRSA with amb. resol.	None	None
OSDL	None	None
MkNN	Error rate	None
MOCA	MAE	None
Stochastic DRSA	None	None
ICT	MAE	None
LPRules	MAE	None
VP-DRSA	None	None
MORE	MAE	None
MPNN MIN–MAX	MSE, error rate	None
VC-bagging	MAE	None
VC-DomLEM	MAE, accuracy	None
REMT	MAE	None
Choquistic regression	Accuracy, AUC	None
VC-DRSA with non-monot. features	Accuracy	None
MC-SVM	Accurary, recall, PPV, NPV, *F*-measure, κ coefficient	FOM
MGain	Accuracy	None
FREMT	Accuracy, MAE	None
MonRF	Accuracy, MAE	NMI
VC-DRSA ORF	None	None
RDMT(H)	Accuracy, κ coefficient, MAE	NMI
RMC-FSVM	Accuracy, recall, PPV, *F*-measure	None
VC-RF	Accuracy, MAE	None
MoNGEL	Accuracy, MAE	NMI
Monot. AdaBoost	Accuracy, MAE	NMI
AntMiner+, cAnt-Miner$_{PB+MC}$	Accuracy	None
EHSMC-CHC	Accuracy, MAE, MAcc, MMAE	NMI
XGBoost	AUC	None
PM-SVM	Accuracy, κ coefficient	MCC
PM-RF	Accuracy	MCC
MMT	Accuracy, MAE	None
FCMT	Accuracy, MAE	None
MCELM	MAE	None
RULEM	Accuracy, MAE, MSE	None
MFARC-HD, FS$_{MOGFS^*}$+T$_{UN^*}$	MAE, MMAE	NMI
MonoBoost	*F*-measure, κ coefficient, recall, accuracy	None
PMDT	Accuracy, MAE	None

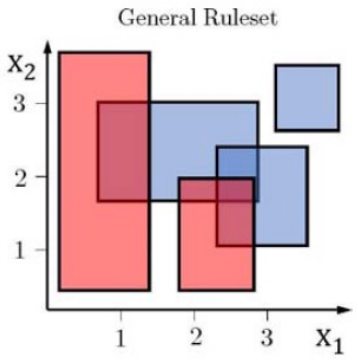
A Comprehensive Taxonomic Overview

Monotonic Classification: Rule and Instance based learning







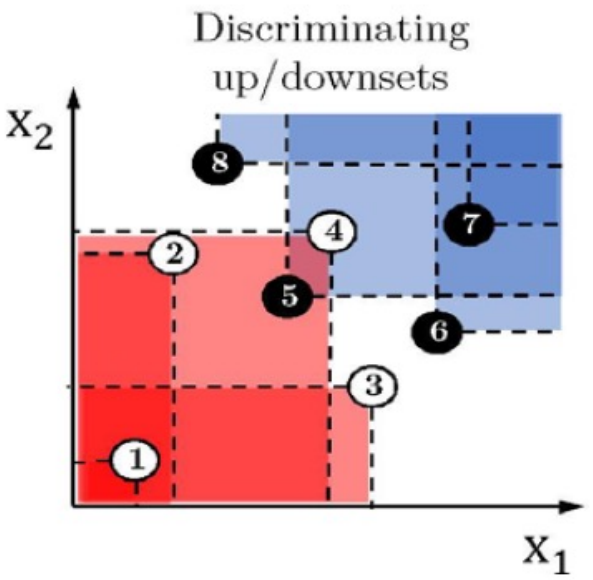


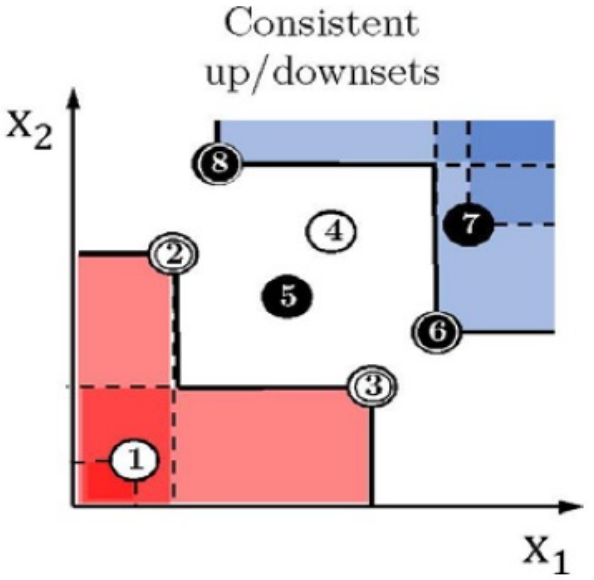
A Comprehensive Taxonomic Overview

Basic Notions

- We define the upset [x]↑ y downset [x]↓ (dominant/dominated sets).

$$[\mathbf{x}]^{\uparrow} = \{\mathbf{x}_i \mid \mathbf{x} \preceq \mathbf{x}_i,\ i = 1..N\}$$
$$[\mathbf{x}]_{\downarrow} = \{\mathbf{x}_i \mid \mathbf{x}_i \preceq \mathbf{x},\ i = 1..N\}$$





Two strategies for converting into a ranking rule:

- Using own points and partial order.
- DRSA: Dominance Rough Sets Approach

A Comprehensive Taxonomic Overview

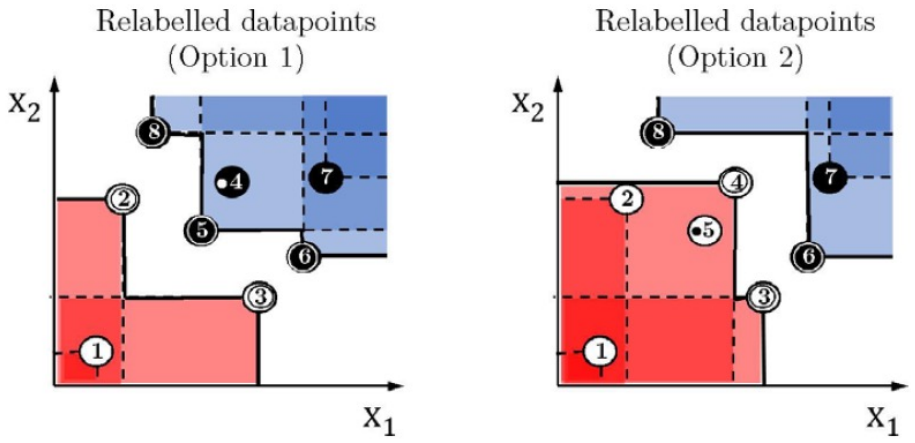
Re-labelling

- There will always be ambiguous areas in non-monotone sets.
- Monotone consistency can be achieved or improved by relabelling. Optimisation problem:

$$\min_{y_i^*} \sum_{i=1}^{N} L(y_i^*, y_i)$$

$$\text{subject to: } \mathbf{x}_i \preceq \mathbf{x}_j \implies y_i^* \leq y_j^*, \quad i, j \in 1..N$$

- Common L loss functions: 0/1 loss, L1 (absolute), L2 (squared)



A Comprehensive Taxonomic Overview

Representative Algorithms: Monotonic k-NN

Two stages:

- The training set is monotonised using an optimal relabelling technique, performing as few relabellings as possible.

- The class label prediction rule of the new examples is modified so that no violations of the monotonicity constraint occur.

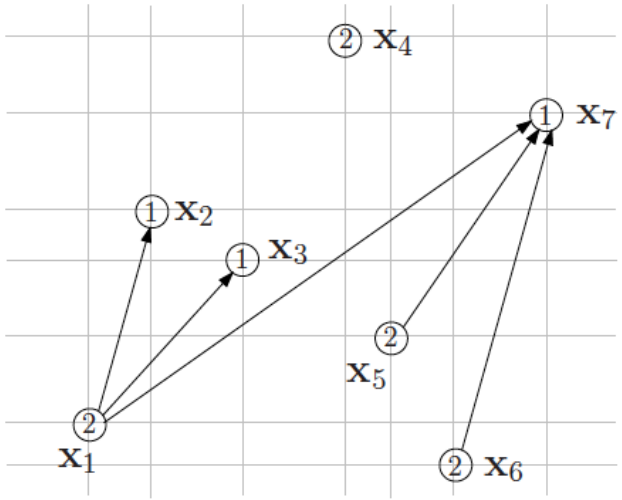


Fig. 1. Example Monotonicity Violation Graph

To satisfy the monotonic constraints, the class label of a new example x0 is constrained to fall in the interval [ymin, ymax], where

$$y_{\min} = \max\{y|(\mathbf{x}, y) \in D \wedge \mathbf{x} \leq \mathbf{x}_0\},$$

$$y_{\max} = \min\{y|(\mathbf{x}, y) \in D \wedge \mathbf{x}_0 \leq \mathbf{x}\},$$

D is a relabelled training set.

Nearest neighbour classification with monotonicity constraints
W Duivesteijn, A Feelders
Machine Learning and Knowledge Discovery in Databases: European Conference (2008)

Representative Algorithms: Monotone SVM

- Existing SVMs add constraints to the original optimisation problem to require the solution to be monotonic for a selected set of points.

$$\mathbf{w}^T \psi(\tilde{\mathbf{x}}_k) \geq \mathbf{w}^T \psi(\mathbf{x}_k), \quad k = 1, \ldots, K$$

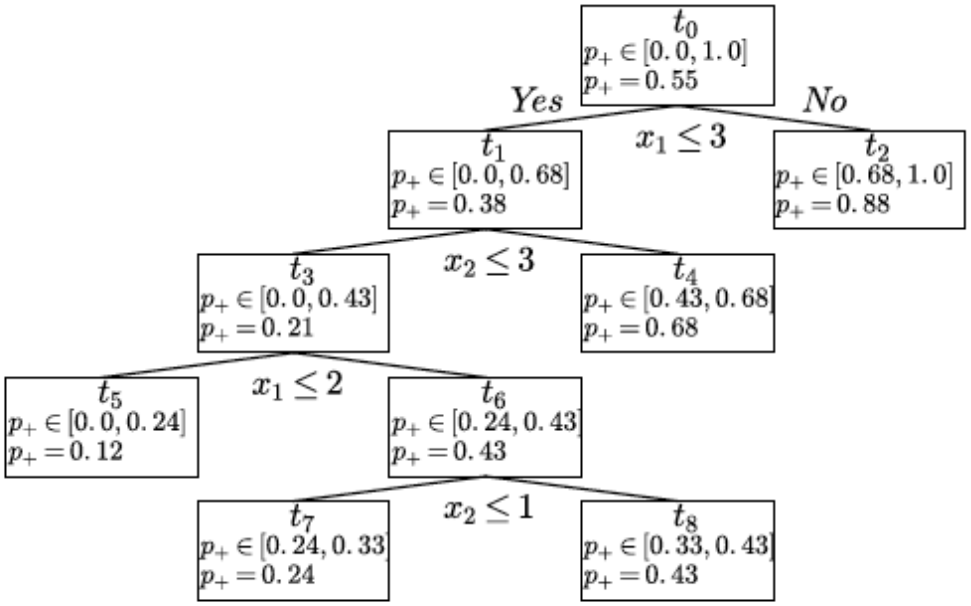
- The *K* constraints are added using two procedures:
  - CJ1: By randomly selecting m=5 training points for each K.
  - CJ2: Partitioning each monotone attribute into K partitions between its minimum and maximum value of the training set. Use k-means per partition to select points.

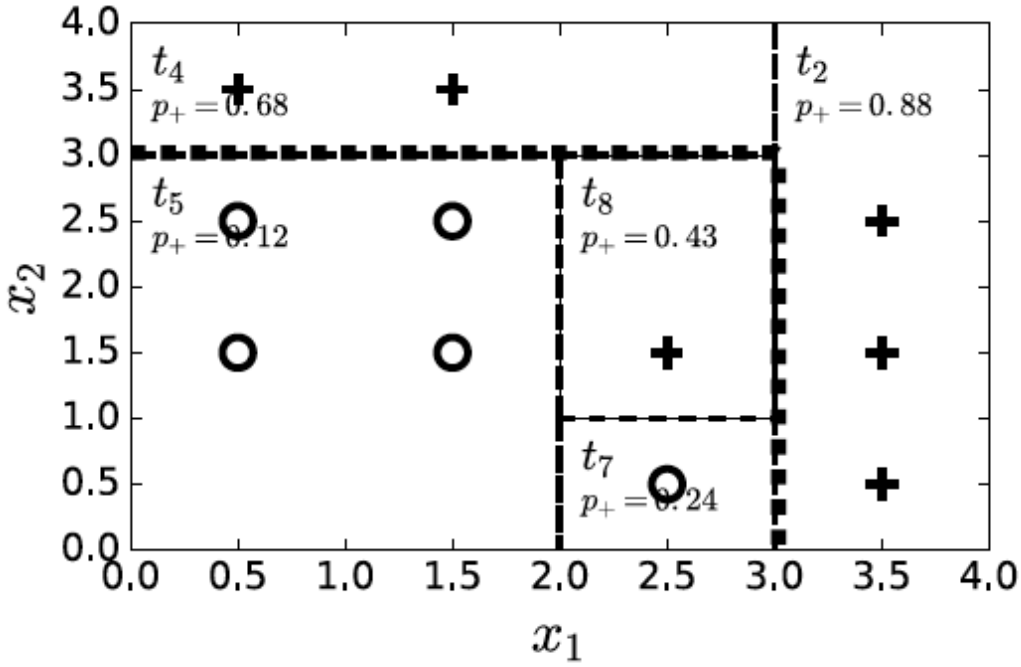
Bartey, Liu and Reynolds 'Effective Monotone Knowledge Integration in Kernel Support Vector Machines (ADMA 2016)

- **Monotone Split Constraint:**
  - The popular gradient boosting library uses this approach.
  - The split constraint is made by adding bounds on coefficients inherited from the leaves.
  - In each split, the mean of the coefficients is passed as an upper bound on future coefficients to the left leaf, and a lower bound to the right leaf.
  - These inherited constraints are respected when finding new splits and guarantee global monotonicity, even though there is no computational burden.
  - The problem is that it greatly deteriorates the accuracy, moving each tree away from the training data.
  - XGBoost only works with binary classes.
  - There are more tree ensembles: FREMT, FCMT, FSD-RF, etc.
  - Arborist package in CRAN or sckit-learn.

A Comprehensive Taxonomic Overview

Representative Algorithms: XGBOOST for Monotonic Classification





- From t_3, it is impossible to predict the class y=+1.
- This results in a failure in t_8, although it could predict p+=0.68 without violating monotonicity.

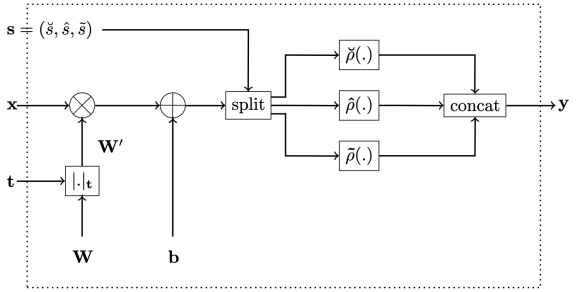
- For SVM or XGBoost it is necessary to use an OVA / OVO.
- The OVA proposal is simple:
  - Variant of Frank and Hall (2001) for Ordinal Regression.
  - Preserves monotonicity and limits L1.
  - $h_c(x) = 1[y \geq c]$, binary classifier distinguishing class less than c from class greater than or equal to c, where 1[condition] = 1 if condition, else 0.
  - The multi-class classifier is given by:

$$h(\mathbf{x}) = 1 + \sum_{c=2}^{C} h_c(\mathbf{x})$$

A Comprehensive Taxonomic Overview

Representative Algorithms: Constrained Monotonic Neural Networks

- Monotonicity by construction:
  - Neural architectures guaranteeing monotonicity (MinMax Networks, Deep Lattice Networks)
- Monotonicity by regularization:
  - Enforcing monotonicity in neural networks during training by employing a modified loss function or a heuristic regularization term.
- Constrained Monotonic NN:
  - Constrained monotone fully connected layer which can be used as a drop-in replacement for a fully connected layer to enforce monotonicity.



Runje and Shankaranarayana. Constrained Monotonic Neural Networks (PMLR 2023)

Preliminaries

A Comprehensive Taxonomic Overview

Some MOC Proposals

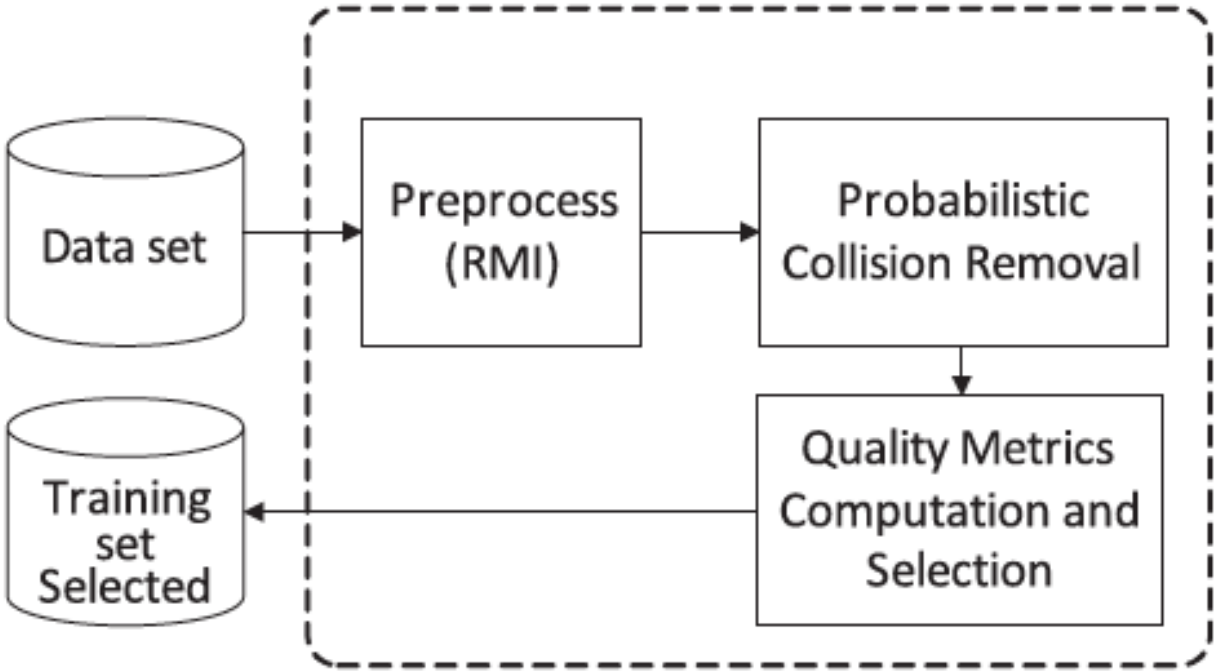
MOC and Fairness in ML

New Paradigms in MC

Conclusions & Future Work

Some MOC Proposals

Training Set Selection



$$\text{Select}x_i = \begin{cases} \text{true} & \textbf{if } \text{Del}(x_i) < \text{Infl}(x_i) \\ & \text{or } \text{Del}(x_i) \geq 0.9 \\ \text{false} & \text{otherwise.} \end{cases}$$

$$\text{Del}(x_i) = \frac{|\text{Dom}(x_i) - \text{NoDom}(x_i)|}{\text{Dom}(x_i) + \text{NoDom}(x_i)},$$

$$\text{Dom}(x_i) = \#X',\ x' \in X' \Leftrightarrow x_i \prec x' \wedge Y(x_i) = Y(x'),$$

$$\text{NoDom}(x_i) = \#Z',\ x' \in Z' \Leftrightarrow x_i \succcurlyeq x' \wedge Y(x_i) = Y(x'),$$

$$\text{nWeight}(x_j) = \frac{\sum_{l=1}^{k} \text{Distance}(x_i, x_l) - \text{Distance}(x_i, x_j)}{\sum_{l=1}^{k} \text{Distance}(x_i, x_l)}, \quad \forall\, x_j \in kNN_{x_i},$$

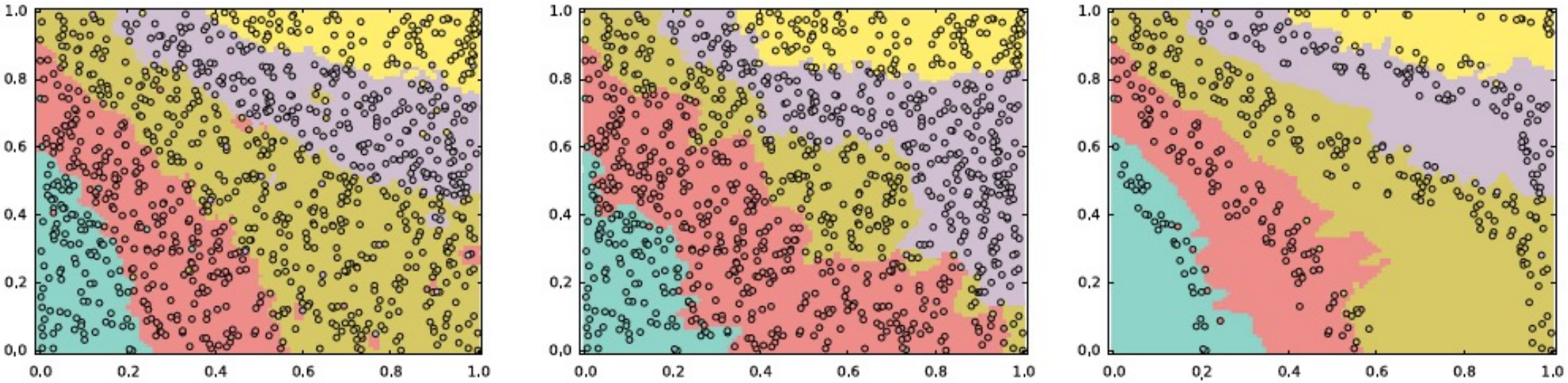
$$\text{influenceWeight}(x_j) = \frac{\text{nWeight}(x_j)}{\sum_{l=1}^{k} \text{nWeight}(x_l)}, \forall\, x_j \in kNN_{x_i},$$

$$\text{Infl}(x_i) = \sum_{j=1}^{k} \text{influenceWeight}(x_j),$$

$$\text{where } Y(x_i) \neq Y(x_j) \wedge x_j \in kNN_{x_i},$$

Some MOC Proposals

Training Set Selection



(a) Artiset Original (b) Artiset Reableled. (c) Artiset MonTSS.

Fig. 3. Artificial data set preprocessed by Relabeling and MonTSS with the borders calculated by M*k*NN with 3 neighbors.

Training set selection for monotonic ordinal classification
JR Cano, S García
Data & Knowledge Engineering 112, 94-105 (2017)

Some MOC Proposals

Monotonic Random Forest

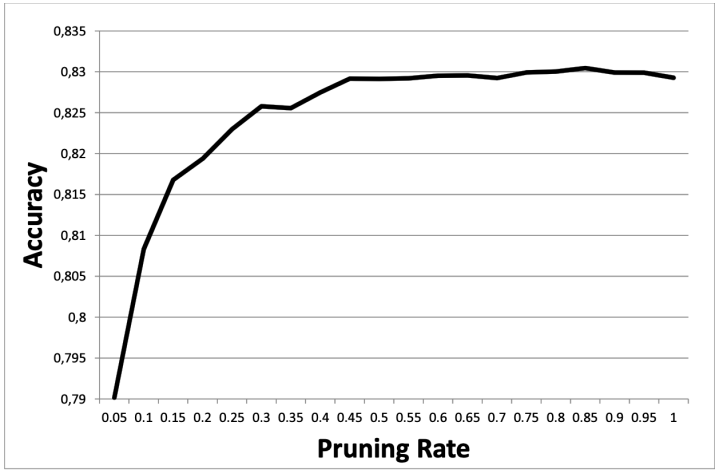
- The R-factor (MID) is used as an extra way to randomise and diversify the different trees constructed in the RF.
- At the same time, we force the tree creation process to be dominated by monotonic considerations.
- To this end, each tree is built from scratch with a different R-factor, chosen with a random number between 1 and Rlimit.
- A pruning mechanism based on a monotonicity index threshold of each resulting tree is used for the final prediction combination.
- Instead of using all trees, the best trees are selected according to the monotonicity violations they generate according to a certain threshold.
- Using the NMI criterion, and sorting the trees in ascending order, the method selects the first *t* trees, where t is a rate in the range (0,1].

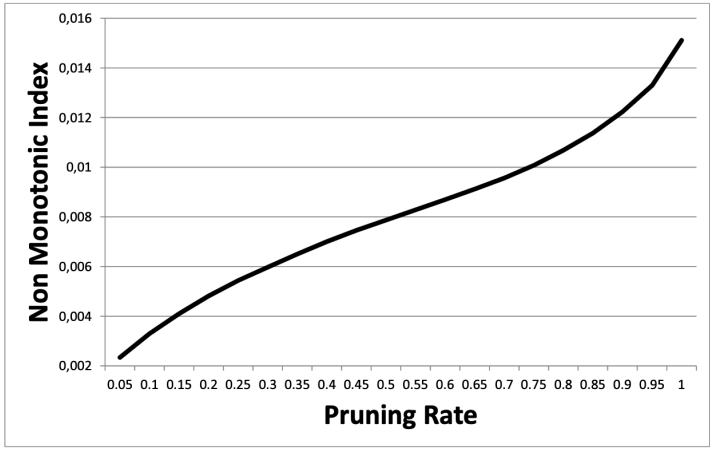
Monotonic random forest with an ensemble pruning mechanism based on the degree of monotonicity
S González, F Herrera, S García
New Generation Computing 33, 367-388 (2015)

Some MOC Proposals

Monotonic Random Forest

- The accuracy is better with a greater number of classifiers.
- The NMI grows with pruning rate: trees are ordered from more to less monotonic.
- There is a clear turning point around the value of 0.5 where the NMI curve starts to grow with a steep slope.





Some MOC Proposals

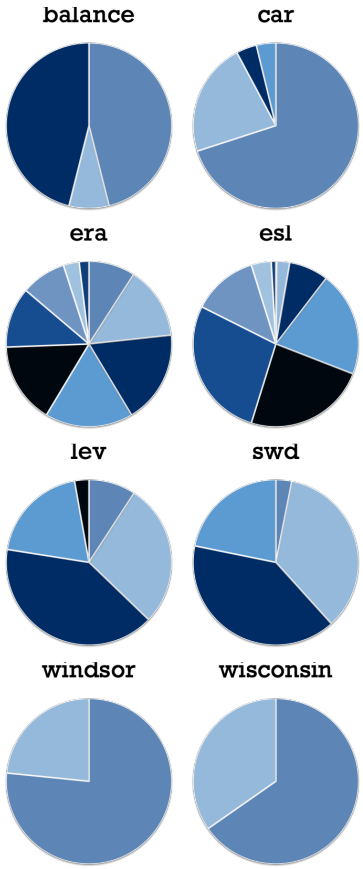
Sampling for monotonic classification

Motivation:

- Many monotonic problems have imbalanced class distributions.
- Sampling techniques are the most convenient solutions, due to their simplicity and the possibility of enabling all monotonic classifiers.
- However, they could worsen the monotonicity of the training set and the models.

Objective (2.II):

- Design sampling techniques for monotonic imbalanced classification that preserve monotonicity and mitigate the impact of skewed distribution.

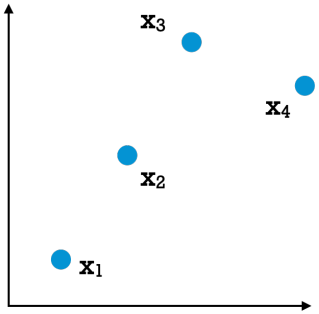


Chain based sampling for monotonic imbalanced classification
S González, S García, ST Li, F Herrera
Information Sciences 474, 187-204
(2019)

Some MOC Proposals

Sampling for monotonic classification

- Monotonic Sampling defines some good sampling practices based on the chains.
- A chain is a set of comparable samples with useful monotonic information. (x_1-x_2-x_3/x_4)
- Three types of instances:
  - Samples at the chain limit: are the greatest or smallest sample in the chains (x_1; x_3 & x_4).
    - Monotonically delimit the class ➟ Preserve or reinforce them.
  - Instances inside chains: are dominated by other instances (x_2: $x_1 < x_2 < x_3$).
    - Less ordering information ➟ Less priority.
  - Monotonic violations: break the monotonicity of the data-sets.
    - Remove or avoid them during Sampling.



These good practices apply to the majority of sampling techniques.
This schema is implemented in 5 popular approaches: RUS, ROS, SMOTE, ADASYN & MWMOTE.

Some MOC Proposals

Sampling for monotonic classification

- **mRUS hierarchically selects** the samples to keep:

If **at-the-limits instances** are enough to balance
then uniformly random selection.
Else-if **inner instances** enough to balance,
then uniformly random selection + previous.
Else selection of **non-monotonic instances** according to the inverse of the number of violations.

- **mROS** selects the instances **according to weights**:
- **At-the-limits Instances** have double weight.
- **Inner samples** have less priority with half the weight.
- If all samples are **non-monotonic violations**, selection according to the inverse of the number of violations.

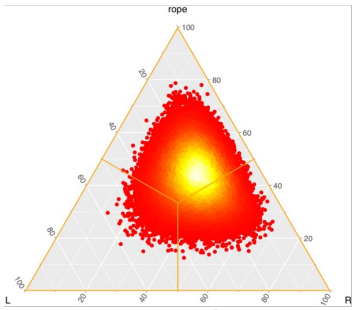
- SMOTE follows the **same scheme** and includes it in the neighbor selection.
- ADASYN and MWMOTE combine this weighting scheme with their original weights.

Some MOC Proposals

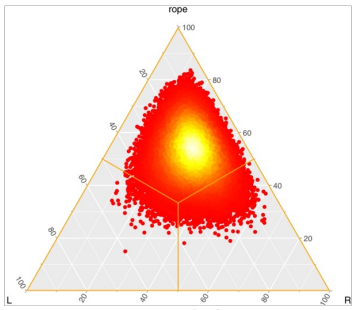
Sampling for monotonic classification

- Monotonic and Standard sampling improves the MAvA performance of the different classifiers.
- There is no significant difference between the standard and monotonic performance in terms of MAvA with Bayesian sign test.

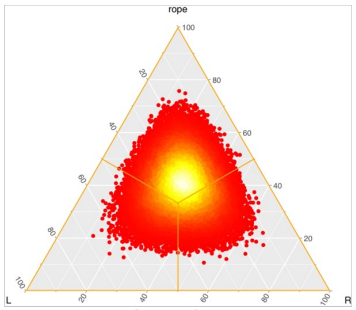
	Original	RUS	mRUS	ROS	mROS	SMOTE	mSMOTE	ADASYN	mADASYN	MWMOTE	mMWMOTE
MkNN	0.5727	0.5827	**0.5834**	0.5742	**0.5907**	0.6106	**0.6190**	0.6143	**0.6189**	0.6128	**0.6160**
OLM	0.5468	0.5273	**0.5679**	**0.5656**	0.5593	**0.5762**	0.5657	**0.5771**	0.5660	**0.5810**	0.5681
MID	0.5434	**0.5693**	0.5580	**0.6272**	0.6053	**0.6064**	0.6015	**0.6104**	0.6054	0.5929	**0.6094**
MonMLP	0.5582	**0.6034**	0.6017	**0.6300**	0.5877	**0.5902**	0.5794	0.5865	**0.5907**	0.5689	**0.5705**
Avg:	0.5553	0.5707	**0.5778**	**0.5993**	0.5857	**0.5959**	0.5914	**0.5971**	0.5953	0.5889	**0.5910**



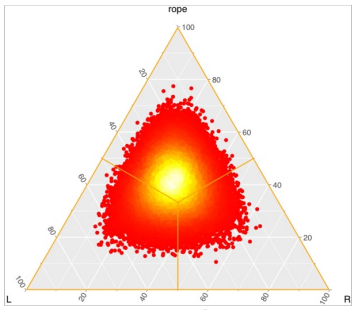
mRUS



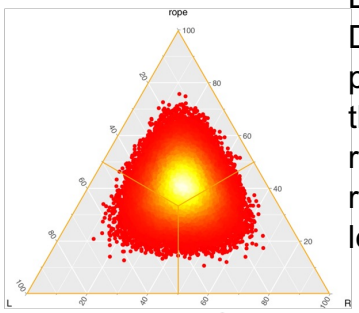
mROS



mSMOTE



mADASYN



mMWMOTE

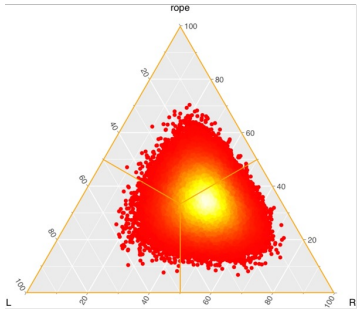
Bayesian sign test: Distribution of probability with their differences. rope: equivalence right: monotonic left: standard

Some MOC Proposals

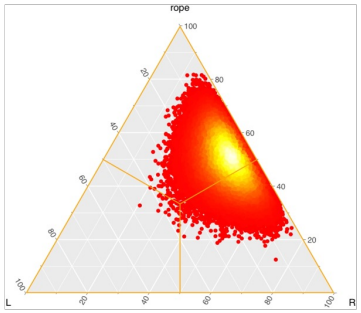
Sampling for monotonic classification

- Monotonic sampling has better NMI than the standard one.
- mRUS and mMWMOTE are significantly better than their standard versions.
- In some cases, Monotonic Sampling improves the NMI of the original classifier.

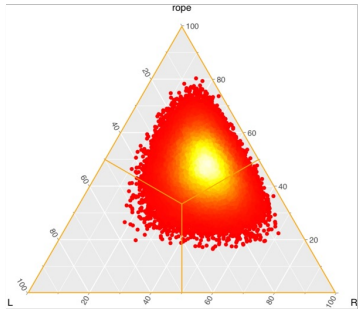
	Original	RUS	mRUS	ROS	mROS	SMOTE	mSMOTE	ADASYN	mADASYN	MWMOTE	mMWMOTE
MkNN	0.0011	0.0034	**0.0023**	0.0014	**0.0013**	0.0020	**0.0019**	0.0024	**0.0019**	0.0020	**0.0019**
OLM	0.0018	0.0028	**0.0017**	**0.0021**	**0.0021**	0.0017	**0.0017**	**0.0016**	0.0017	0.0019	**0.0018**
MID	0.0030	0.0090	**0.0074**	0.0050	**0.0028**	0.0038	**0.0027**	0.0045	**0.0030**	0.0042	**0.0030**
MonMLP	0.0005	0.0021	**0.0020**	0.0013	**0.0010**	0.0013	**0.0012**	0.0015	**0.0012**	0.0036	**0.0013**
Avg:	0.0016	0.0043	**0.0034**	0.0025	**0.0018**	0.0022	**0.0019**	0.0025	**0.0020**	0.0029	**0.0020**



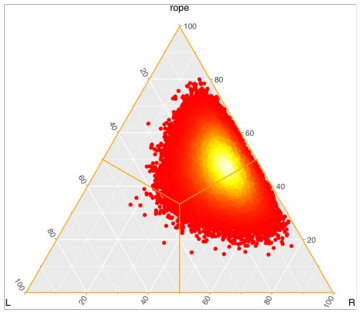
mRUS



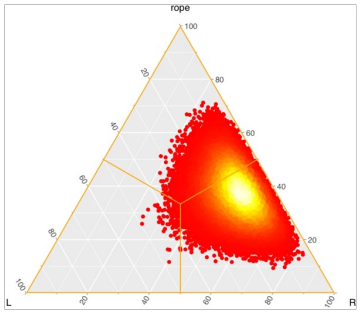
mROS



mSMOTE



mADASYN



mMWMOTE

Some MOC Proposals

Monotonic Fuzzy kNN

Monotonic FkNN has two different stages:

1.Preparation: extraction of the class memberships of the training samples.

- Reduce the relevance of monotonic noise or fix them.

2.Prediction: aggregation of the memberships of the neighbors.

- Give flexibility between monotonicity and accuracy.

Both stages use Monotonic kNN that restricts the neighbors to the monotonic valid classes of the evaluated sample x.

Fuzzy k-nearest neighbors with monotonicity constraints: Moving towards the robustness of monotonic noise
S González, S García, ST Li, R John, F Herrera
Neurocomputing 439, 106-121 (2019)

Monotonic Fuzzy kNN

- First, repeated instances with different labels are combined into one class membership with their frequencies of each class:

$$u(x, l) = \frac{|\{x_j \in \mathcal{D} \,|\, x_j = x \wedge y_j = l\}|}{|\{x_j \in \mathcal{D} \,|\, x_j = x\}|}$$

- The memberships of the remaining samples are estimated with a Monotonic kNN using the medians of previous memberships:
  - Given *x*, its neighbors are restricted to its monotonic valid classes.
  - Following this aggregation, where *RCr* is the minimum membership for the original label.

$$u(x_i, l) = \begin{cases} RCr + (nn_l/k) * (1 - RCr) & \text{if} \quad y_i = l \\ (nn_l/k) * (1 - RCr) \end{cases}$$

Some MOC Proposals

Monotonic Fuzzy kNN

- Prediction stage monotonically restricts the aggregation with MkNN (inRange) or contribution penalty - *pOR* (outRange).
- Valid classes are defined with the medians from the previous memberships.

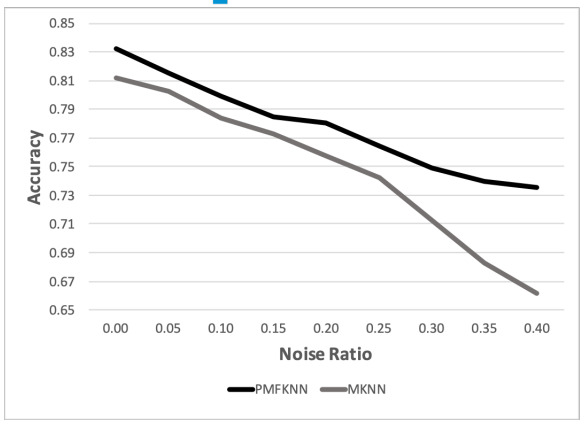
$$u(x,l) = \frac{\sum_{j=1}^{K} u(x_j,l) * \frac{pOR_j}{||x - x_j||^{(m-1)}}}{\sum_{j=1}^{K} \frac{pOR_j}{||x - x_j||^{(m-1)}}}$$

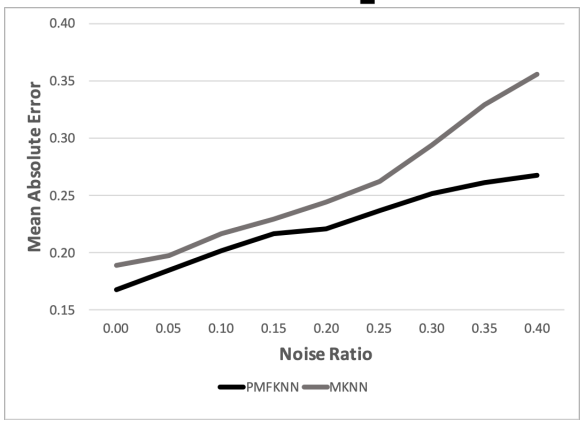
- Two distinct versions:
  - Pure Monotonic (PM): *RCr = 0.5 & inRange*
  - Approximate Monotonic (AM): *RCr = 1 & outRange (pOR = 0.5)*

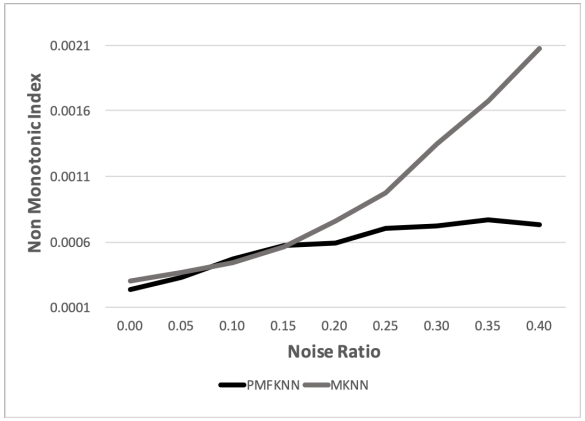
Some MOC Proposals

Monotonic Fuzzy kNN

- We gradually introduce noisy samples into Artiset by changing the labels of some samples according to their adjacent classes.
- MonFkNN is always better than MkNN for all metrics.
- Their differences increase with higher noise ratios.
- The slope of deterioration of MonFkNN performance remains more stable.







Some MOC Proposals

Metric Learning for Monotonic Classification

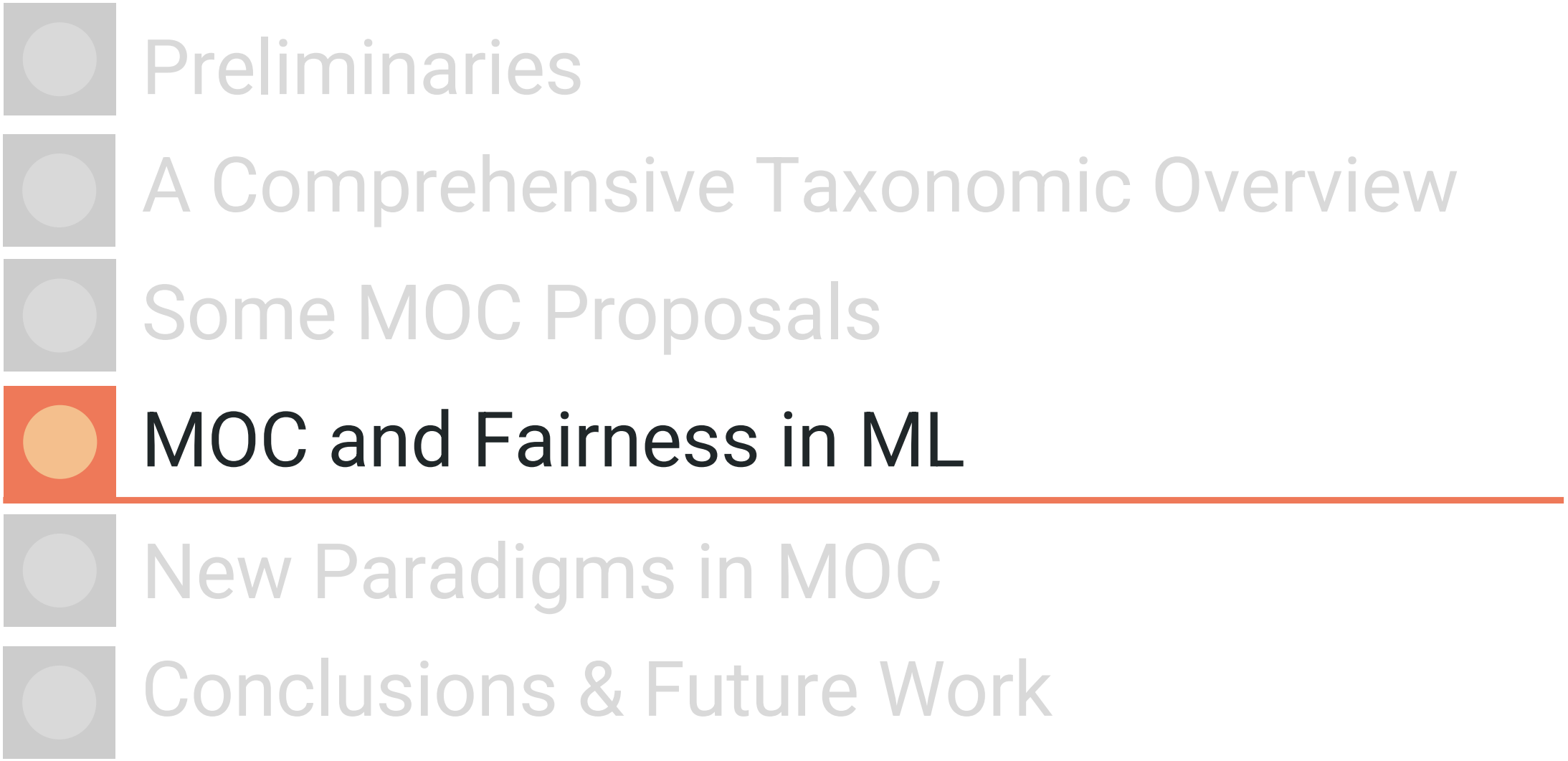
- The performance of distance-based classifiers can be improved with the use of distance metric learning algorithms, which are able to find the distances that best represent the similarities among each pair of data samples.

- However, learning a distance for monotonic data has an additional drawback, as the learned distance may negatively perturb the monotonic constraints of the data.

- We propose a new model for learning distances that does not corrupt these constraints. This methodology will also be useful in identifying and discarding non-monotonic pairs of samples that may be present in the data due to noise.

- We explote the linear tranformations to reduce the non-monotonicity of the data set, using M-matrices.

Metric Learning for Monotonic Classification: Turning the Space up to the Limits of Monotonicity
JL Suárez, G González-Almagro, S García, F Herrera
To appear in Applied Intelligence


Preliminaries
A Comprehensive Taxonomic Overview
Some MOC Proposals
MOC and Fairness in ML
New Paradigms in MOC
Conclusions & Future Work

MOC and Fairness in ML

AI ethics and safety

Monotonicity: When the value of the predictor changes in a given direction, the value of the response variable changes consistently either in the same or opposite direction.

The interpretable prediction yielded by the model can thus be **directly inferred.**

This monotonicity dimension is also a **highly desirable** interpretability condition of predictive models in many heavily regulated sectors, because it incorporates reasonable expectations about the consistent application of sector specific selection constraints into automated decision-making systems.

MOC and Fairness in ML

AI ethics and safety

- Apart from the requirements of having models with high accuracy, there is also a need for transparency and interpretability, and **monotonicity helps in partially achieving the above**.
- Due to legal, ethical and/or safety concerns, monotonicity of predictive models with respect to some input or all the inputs is required in numerous domains such as financial (house pricing, credit scoring, insurance risk), healthcare (medical diagnosis, patient medication) and legal (criminal sentencing) to list just a few.
- For example, when using machine learning to predict admission decisions, it may seem unfair to select student X over student Y, if Y has a higher score than X, while all other aspects of the two are identical.
- A model without such a monotonic property would not, and certainly should not, be **trusted by society to provide a basis for such important decisions.**

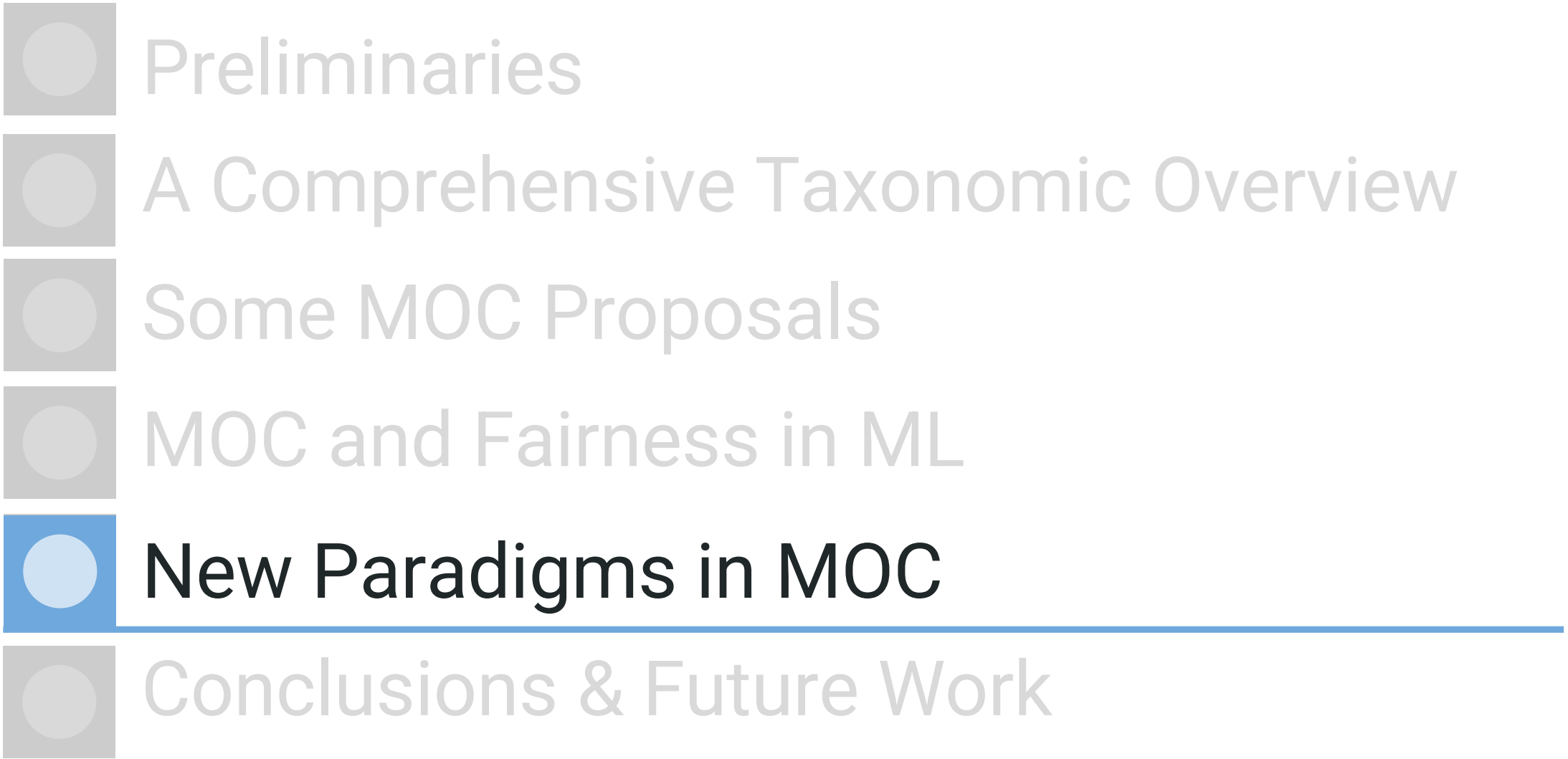
Gupta, M., Cotter, A., Pfeifer, J., Voevodski, K., Canini, K., Mangylov, A., Moczydlowski, W., and Van Esbroeck, A. Monotonic calibrated interpolated look-up tables. The Journal of Machine Learning Research, 17(1):3790–3836, 2016

MOC and Fairness in ML

Fairness in AI

- While these two concepts address different aspects of machine learning models, they can be related when we consider the fairness of a machine learning model under monotonicity constraints.
- When considering fairness in AI, these monotonicity constraints **can be useful in addressing bias and discrimination** in classification algorithms. By imposing constraints that enforce fairness, we can ensure that the model's predictions do not unfairly favor or discriminate against certain individuals or groups based on sensitive attributes such as gender, race, or age.
- One approach to incorporating fairness constraints in classification models is through the use of **counterfactual fairness**. Counterfactual fairness aims to ensure that the model's predictions do not change when the sensitive attributes are changed while keeping other features fixed. This can be achieved by formulating the **fairness constraints as optimization problems** and incorporating them into the training process of the classification model.

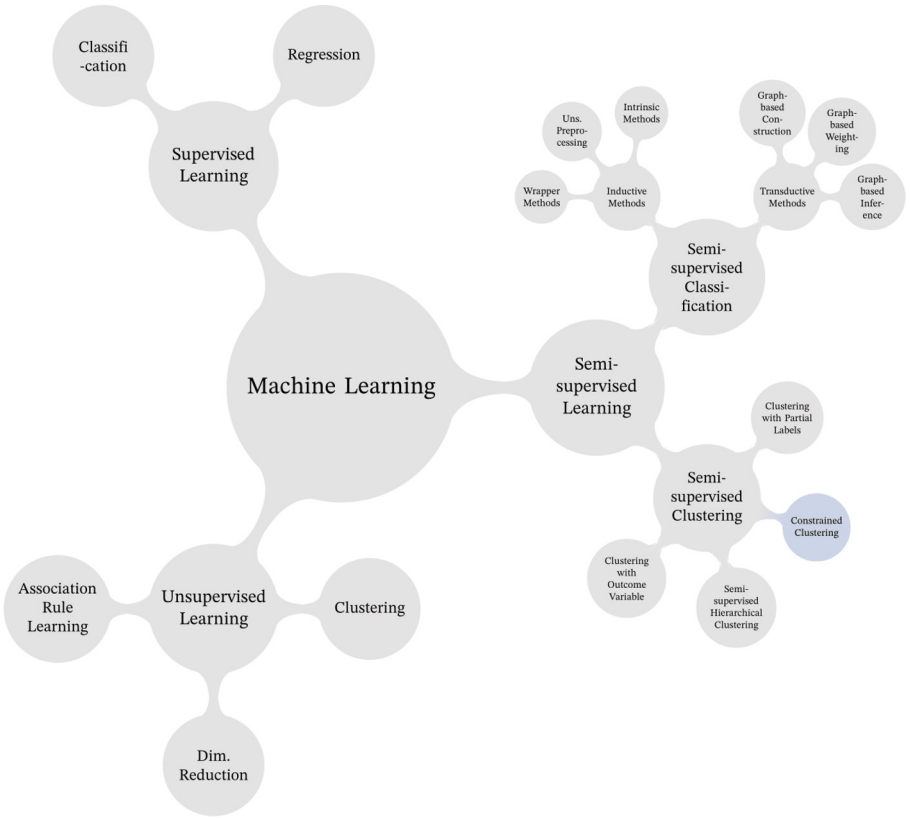
Fairness Constraints: A Flexible Approach for Fair Classification
Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, Krishna P. Gummadi; 20(75):1–42, 2019.


Preliminaries
A Comprehensive Taxonomic Overview
Some MOC Proposals
MOC and Fairness in ML
New Paradigms in MOC
Conclusions & Future Work

New Paradigms in MOC

Semi-Supervised Clustering

Semi-Supervised Learning: tries to combine the benefits of supervised and unsupervised learning by making use of both labeled and unlabeled data, or **other kinds of expert knowledge.**

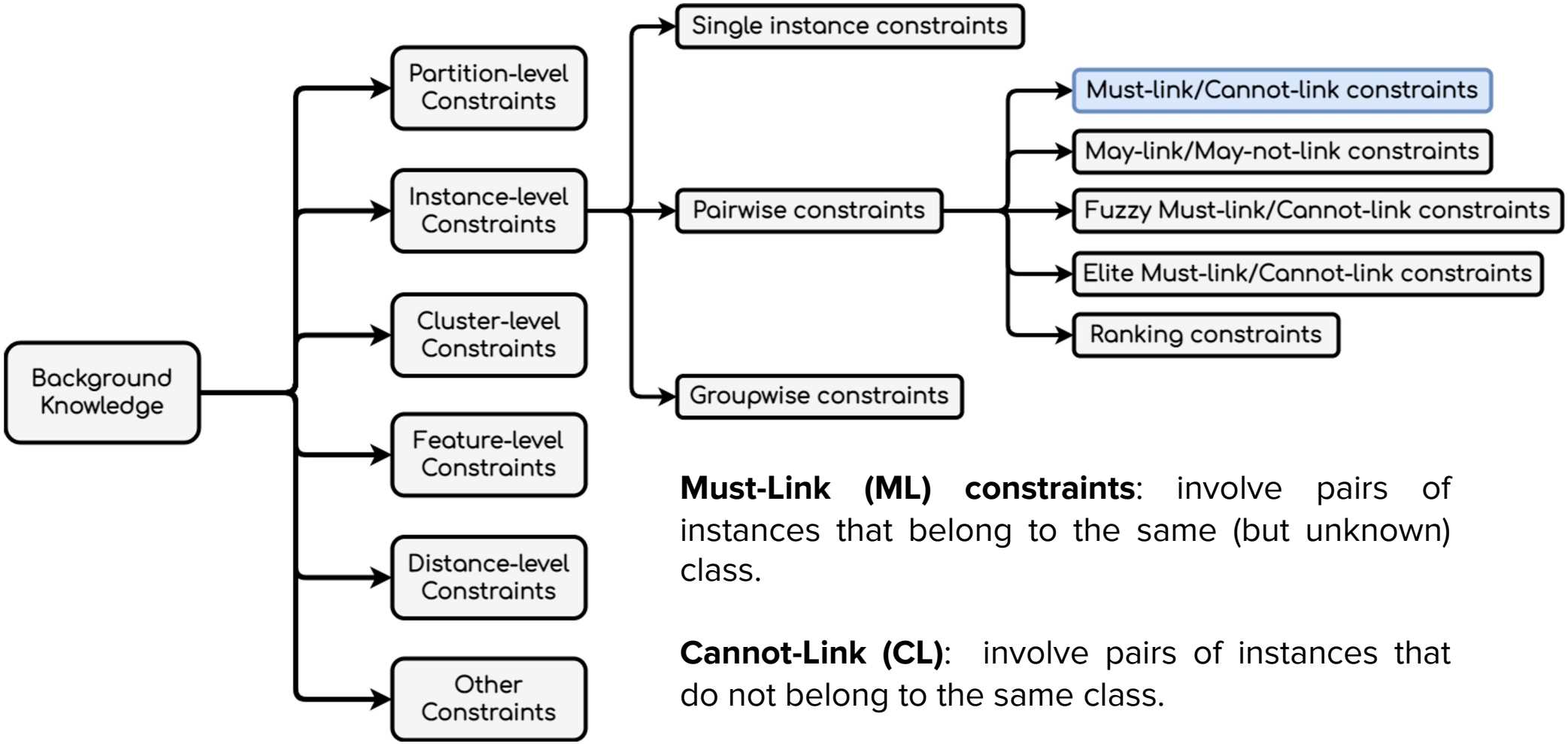


Semi-Supervised Clustering: in addition to the unlabeled dataset, background knowledge is given to perform clustering. When the **background knowledge is given in the form of constraints**, the resulting clustering paradigm is known as **Constrained Clustering (CC).**

Semi-supervised Clustering with Two Types of Background Knowledge: Fusing Pairwise Constraints and Monotonicity Constraints
Germán Gonzalez-Almagro, Pablo Sánchez-Bermejo, Juan Luis Suárez, José Ramón Cano, Salvador García. To appear in Informatiom Fusion.

New Paradigms in MOC

Constrained Clustering



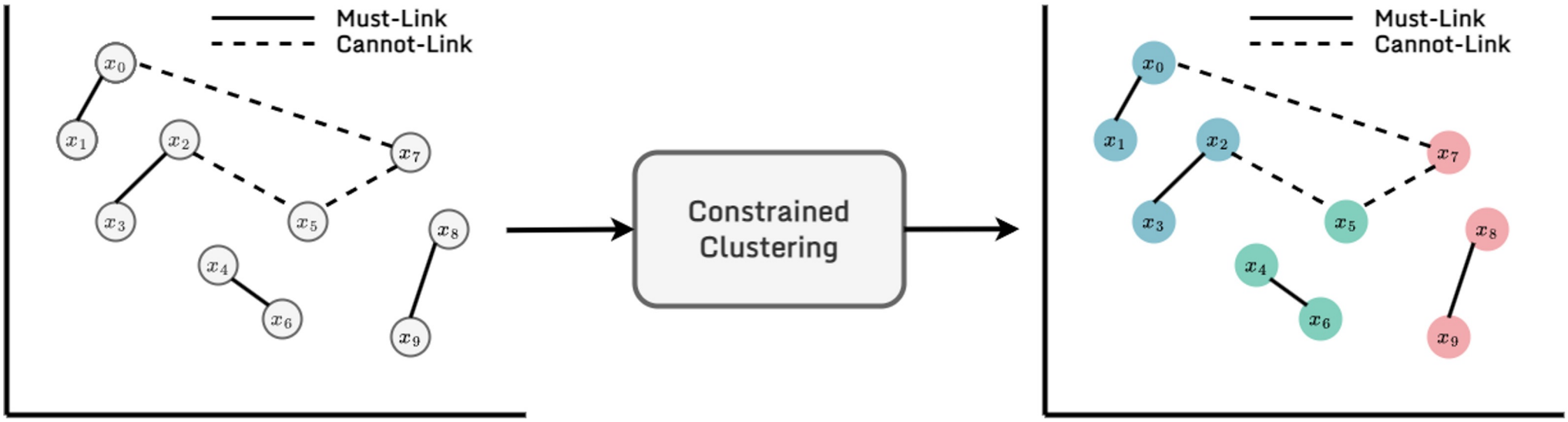
Must-Link (ML) constraints: involve pairs of instances that belong to the same (but unknown) class.

Cannot-Link (CL): involve pairs of instances that do not belong to the same class.

New Paradigms in MOC

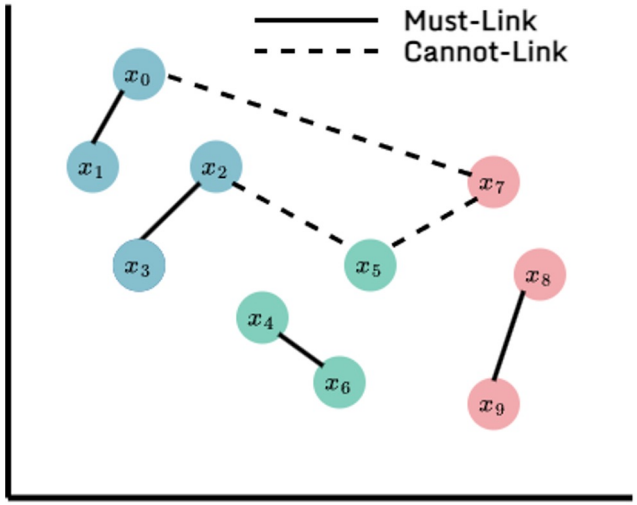
Instance-level Pairwise Constrained Clustering (CC)

The goal of constrained clustering is to **find a partition of the dataset that ideally meets all constraints** in the union of both constraint sets.

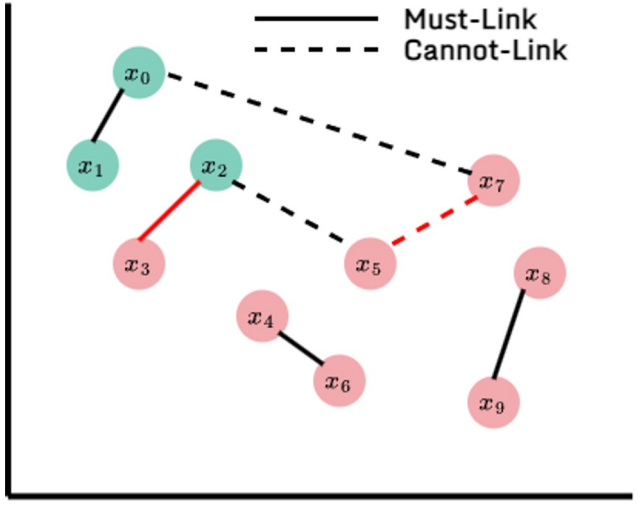


The Infeasibility

The infeasibility refers to **the number of constraints broken** by a given partition.



$\mathbf{Infs}(C, CS) = 0$

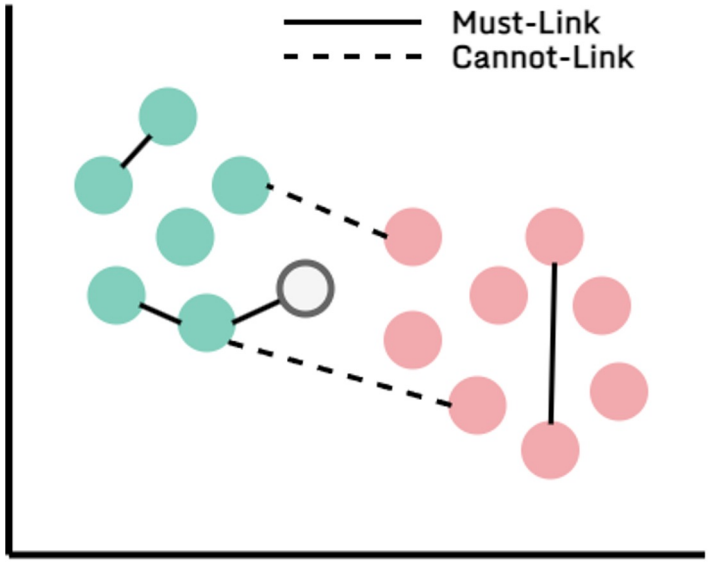


$\mathbf{Infs}(C, CS) = 2$

$$\mathrm{Infs}(C, CS) = \sum_{C_{=}(x_i, x_j) \in CS} \mathbb{1}[\![l_i^C \neq l_j^C]\!] + \sum_{C_{\neq}(x_i, x_j) \in CS} \mathbb{1}[\![l_i^C = l_j^C]\!]$$

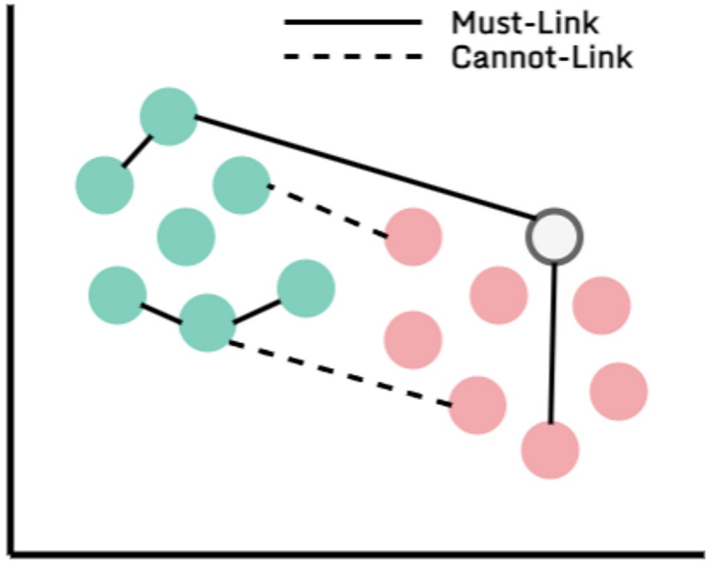
Hard Constraints VS. Soft Constraints

Hard Constraints



¿Infs$(C, CS) = 0$?

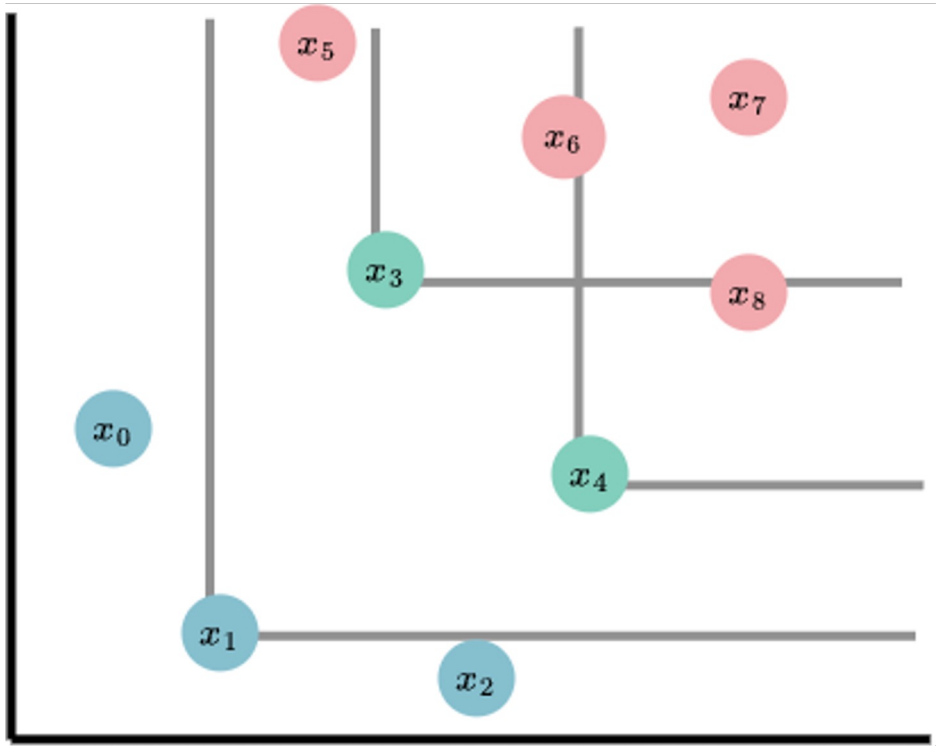
Soft Constraints



¿Infs$(C, CS) = 0$? ¿Infs$(C, CS) = 1$?

Soft constraints are **resilient to noise**, and allow for **flexibility** in the cost/objective function and in the optimization procedure.

MCDA introduces the concept of preference. The **preference** quantifies the addition of differences between the features of two instances, **limited to the features in which one of them is strictly better than the other**.



$$r(x_i, x_j) = \sum_{d : x_{[i,d]} > x_{[j,d]}}^{u} w_d x_{[i,d]} - w_d x_{[j,d]}.$$

$$L_1(x_i, x_j) = r(x_i, x_j) + r(x_j, x_i).$$

PCKM-Mono – Objective Function

An instances is assigned to a cluster if **its centroid is the most similar to it in terms of preference**, taking the number of violated constraints into account.

$$J_{PCKMM} = \frac{1}{K}\sum_{k=1}^{K}\sum_{x_i \in c_k} \boxed{|(r(x_i,\mu_k) - r(\mu_k,x_i)|} + \sum_{(x_i,x_j)\in C_=} \mathbb{1}[\![l_i \neq l_j]\!] + \sum_{(x_i,x_j)\in C_\neq} \mathbb{1}[\![l_i = l_j]\!]$$

PCKM-Mono – Objective Function

An instances is assigned to a cluster if **its centroid is the most similar to it in terms of preference**, taking the number of violated constraints into account.

$$J_{PCKMM} = \frac{1}{K}\sum_{k=1}^{K}\sum_{x_i \in c_k} |(r(x_i,\mu_k) - r(\mu_k,x_i)| + \boxed{\sum_{(x_i,x_j)\in C_=} \mathbb{1}[\![l_i \neq l_j]\!] + \sum_{(x_i,x_j)\in C_\neq} \mathbb{1}[\![l_i = l_j]\!]}$$

PCKM-Mono – Expectation-Minimization

An instances is assigned to a cluster if its centroid is the most similar to it in terms of preference, taking the number of violated constraints into account.

$$\operatorname{argmin}_h \Big(\Big| \sum_{j=1}^{u} (x_{[i,j]} - \mu_{[h,j]}) \Big| + \sum_{x_j:(x_i,x_j)\in C_=} \mathbb{1}[\![l(c_h) \neq l_j]\!] + \sum_{x_j:(x_i,x_j)\in C_\neq} \mathbb{1}[\![l(c_h) = l_j]\!] \Big)$$

The **centroids must be neutral** in terms of preference with respect to all instances in the cluster.

$$\mu_i = \frac{1}{|c_i|} \sum_{x_i \in c_i} x_i$$

New Paradigms in MOC

Datasets with monotonicity constraints

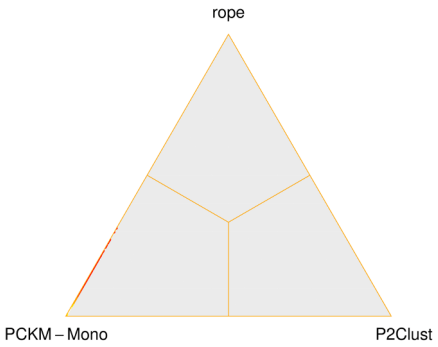
Dataset	Instances	Classes	Features	CS_{10}		CS_{15}		CS_{20}	
				ML	CL	ML	CL	ML	CL
Artiset	899	10	2	494	3422	1240	7671	2061	13870
Balance	625	3	4	832	1059	1799	2479	3332	4418
BostonHousing4CL	506	4	13	284	941	686	2089	1266	3784
Car	1728	4	6	7961	6745	18167	15244	32076	27264
ERA	1000	9	4	676	4274	1562	9613	2760	17140
ESL	488	9	4	216	912	521	2107	949	3707
LEV	1000	5	4	1381	3569	3174	8001	5692	14208
MachineCPU	209	4	6	41	149	99	366	205	615
Qualitative Bankruptcy	250	2	6	35	147	153	344	322	617
SWD	1000	4	10	1566	3384	3674	7501	6583	13317
Windsor Housing	546	2	11	915	516	2105	1135	3827	2059
Wisconsin	683	2	9	1273	1005	2834	2317	5146	4034

New Paradigms in MOC

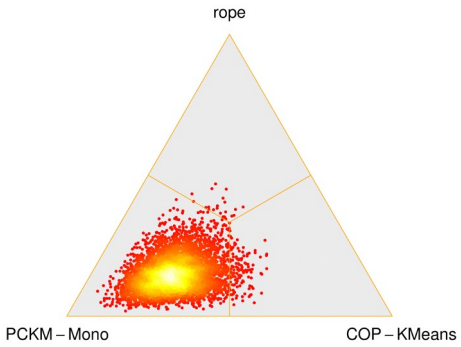
Validation – ARI

PCKM-Mono represents a statistically significant improvement over all compared method with respect to ARI, except for the comparison against PCSKMeans, which is the most debated one, with a slight advantage for PCSKMeans.

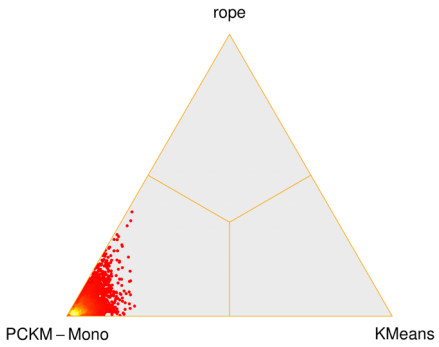
P2Clust



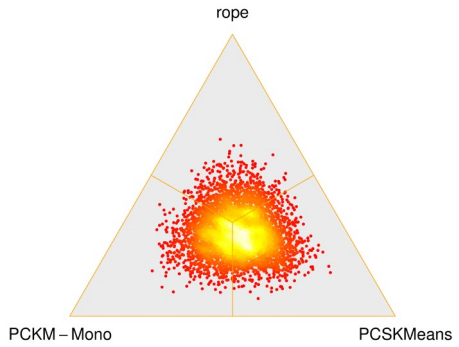
COP-Kmeans



KMeans



PCSKMeans

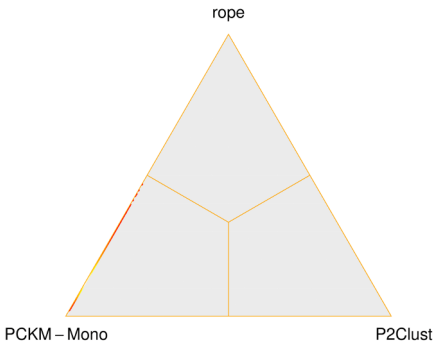


New Paradigms in MOC

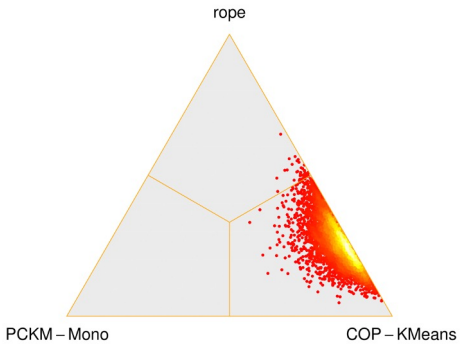
Validation – NMI

NMI measures the degree in which monotonicity is broken by a partition. Indisputable superiority of purely monotonic algorithms with respect to NMI, but convincing results with respect to the rest.

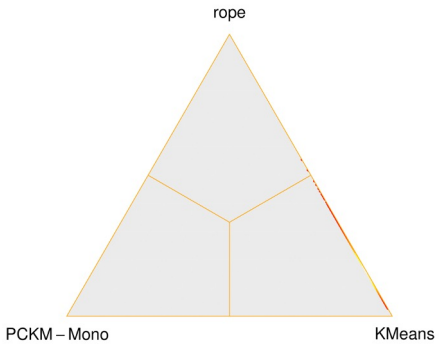
P2Clust



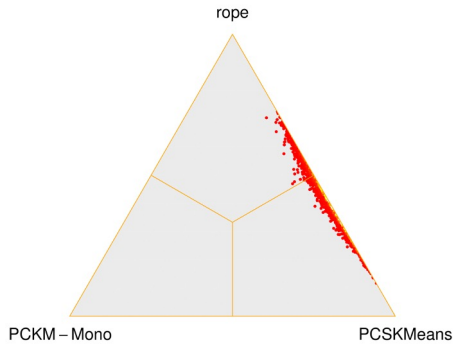
COP-Kmeans



KMeans



PCSKMeans

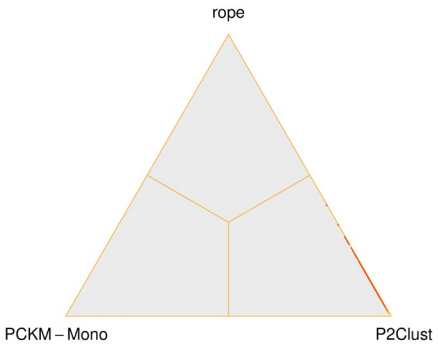


New Paradigms in MOC

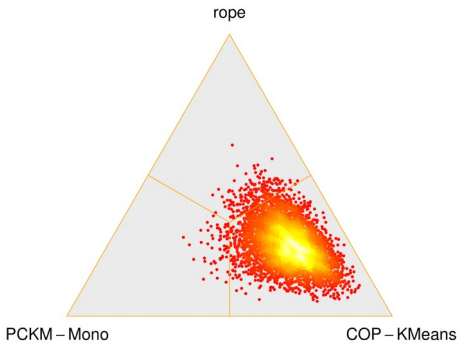
Validation – Unsat

Unsat measures the degree in which pairwise constraints are broken by a partition. PCKM-Mono represents an advantage over all other proposals, except for PCSKMeans, for which no statistically significant differences are found.

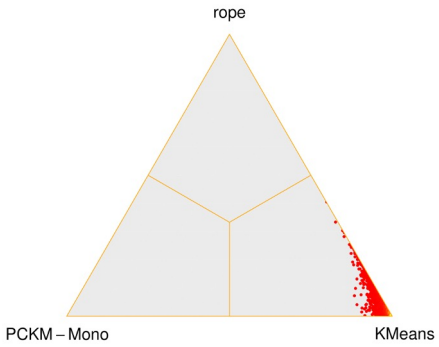
P2Clust



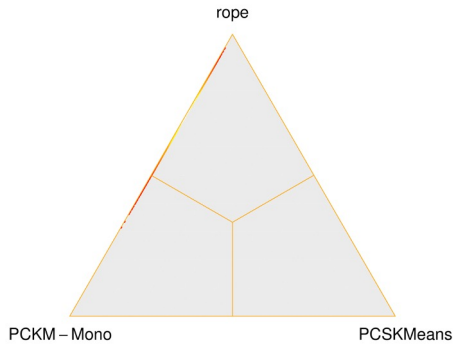
COP-Kmeans



KMeans

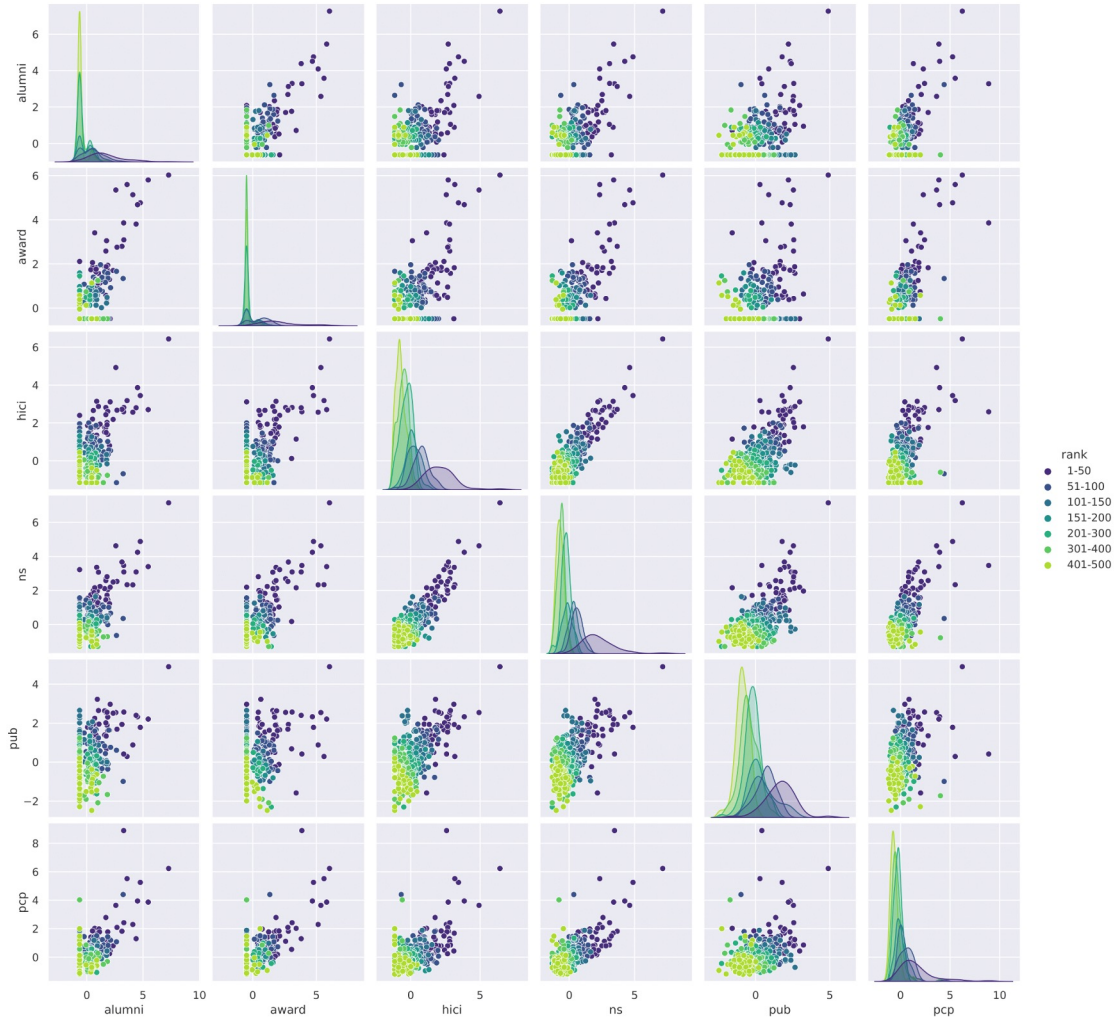


PCSKMeans



New Paradigms in MOC

Case of Study



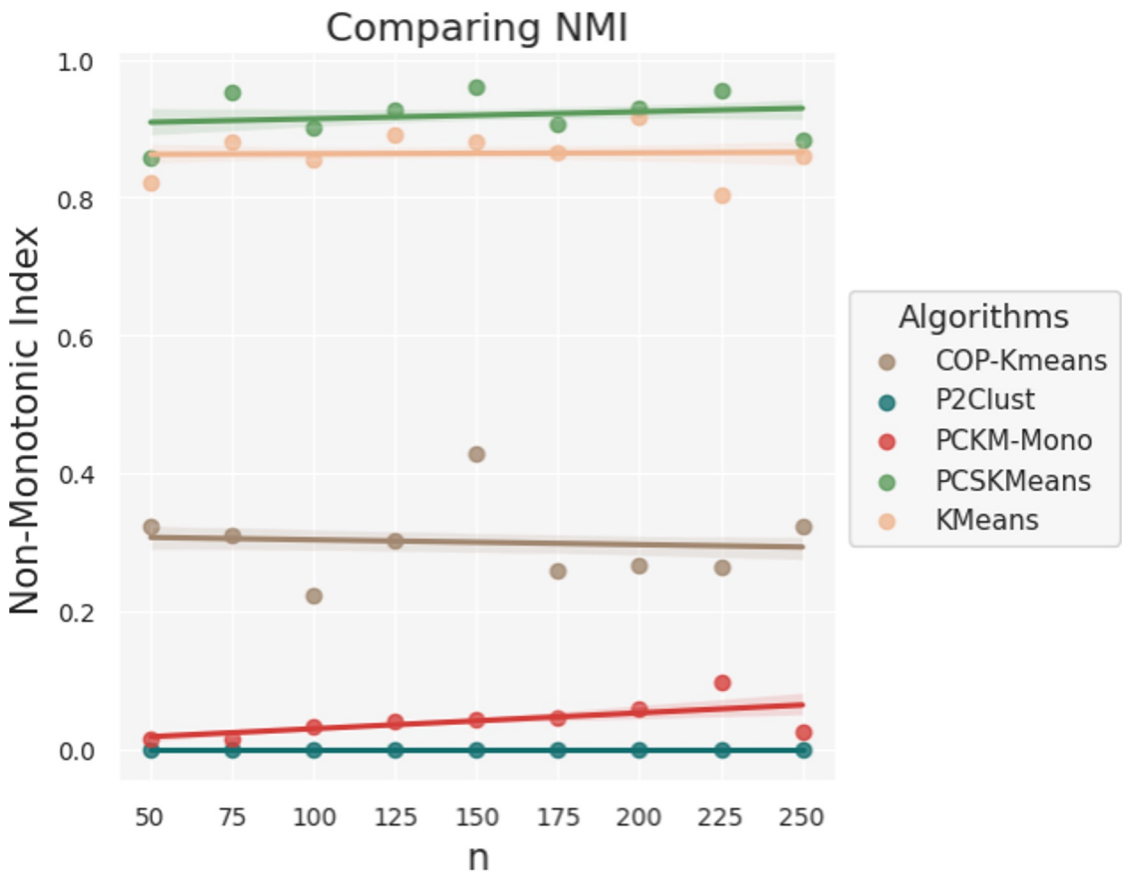
Shanghai Ranking of World Universities

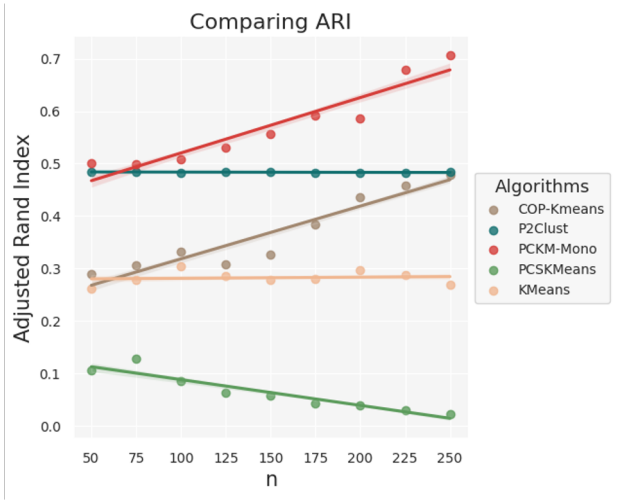
Clustering problem with both monotonicity constraints and instance-level pairwise constraints.

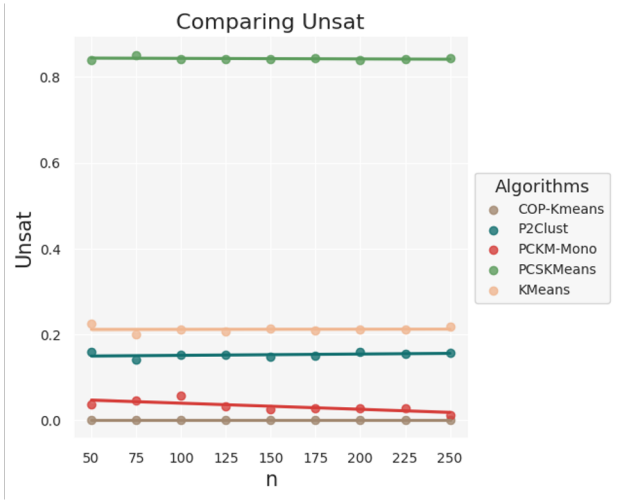
Features monotonicity faults in 7% of its instances.

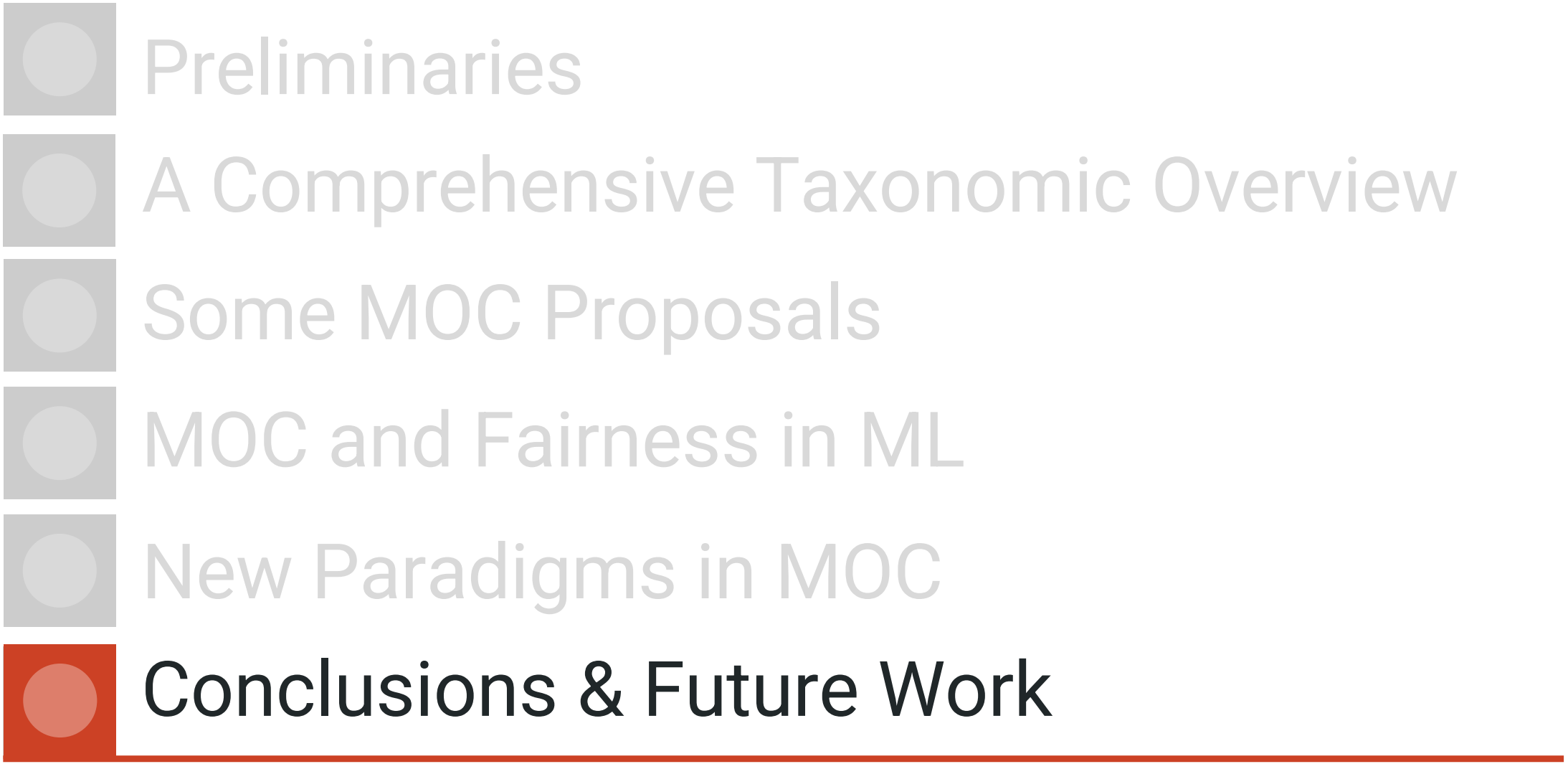
New Paradigms in MOC

Case of Study








Preliminaries
A Comprehensive Taxonomic Overview
Some MOC Proposals
MOC and Fairness in ML
New Paradigms in MOC
Conclusions & Future Work

Conclusions

- Monotonicity constraints can help **ensure the model's output** is in line with the expected trend, they may not be appropriate for all scenarios. It's important to **understand the nature of the data** and the specific requirements of the task before deciding to apply these constraints.

- Monotonicity constraints can be used to **improve the interpretability and practicality** of machine learning models.

- Fairness and Monotocity are constraints which are **highly related** in modern AI learning systems.

Conclusions & Future Work

Future Work

- **Creation of a free MOC library**. The creation of an open-access library specialized in MOC methods would greatly stimulate research in the area.
- **Monotonicity Constraints in Other Models.** Especially in Neural Networks, although there are more models that require to be adapted to handle this scenario (Bayesian Networks, Oblique Decision Trees, OVO decomposition).
- **Handling Non-Linearity.** Future research could focus on developing methods that allow models to capture non-linear relationships while still respecting the monotonicity constraints.
- **Advanced Techniques for Enforcing Monotonicity.** Current methods may not always ensure strict monotonicity, especially in the presence of noisy data.
- **Evaluation of Monotonic Models.** Developing robust methods for evaluating the performance of models with monotonicity constraints. This could include developing new metrics or evaluation strategies that take into account the monotonic nature of the model.
- **New combinations of types of background knowledge**. Another potential research direction is to investigate how to automatically identify the best combination of background knowledge for a given problem, not only monotonicity, hence keeping low human effort and cost.

Monotonic Ordinal Classification

A Path to Fairness in Machine Learning Prediction

Thank you!