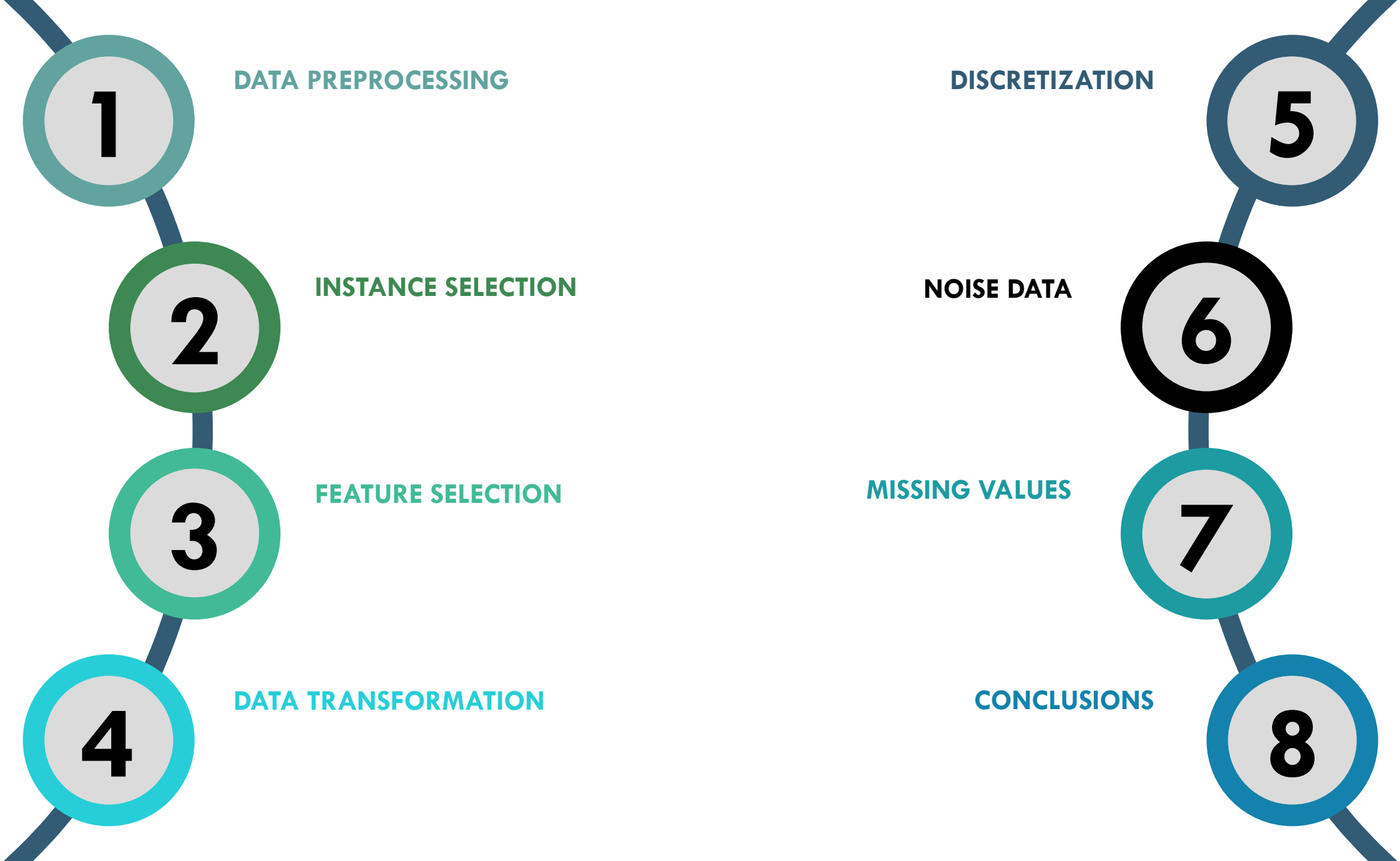
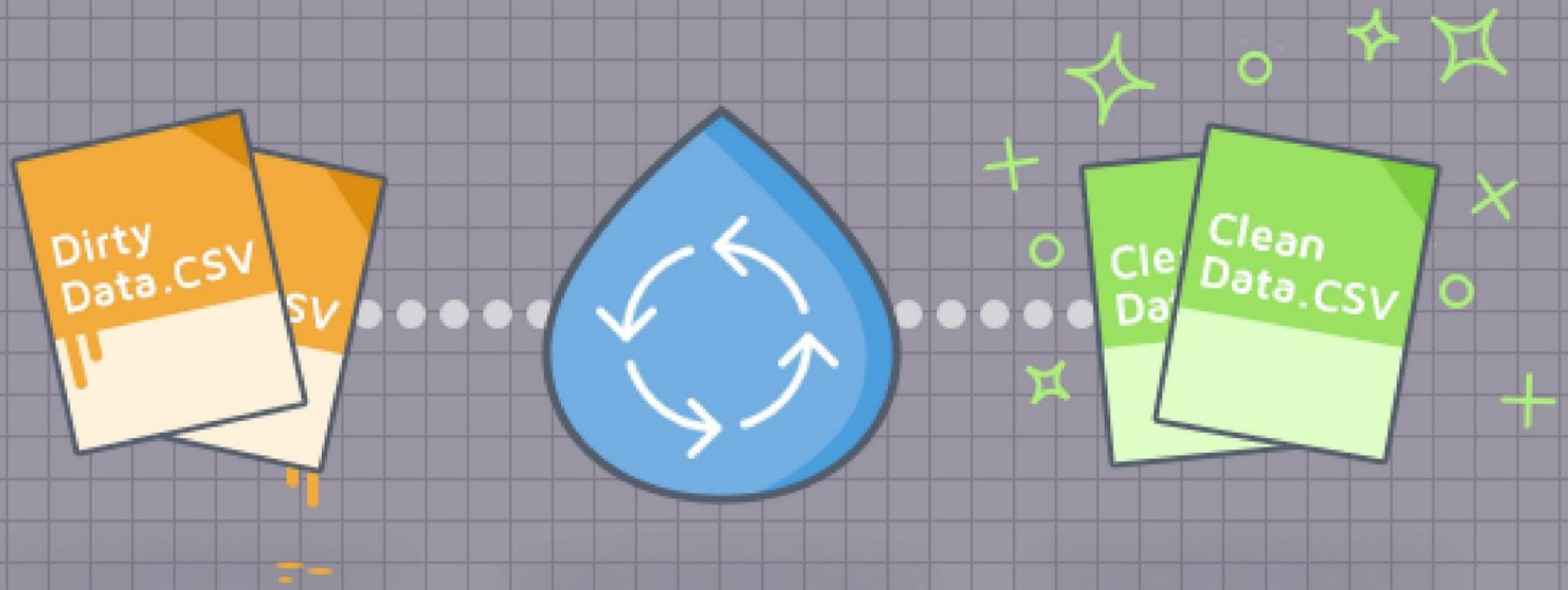
DATA PREPROCESSING IN ML

SOMACHINE 2020| IAA-CSIC + UGR
November 24th 2020

IAA INSTITUTO DE ASTROFÍSICA DE ANDALUCÍA
EXCELENCIA SEVERO OCHOA
CSIC
DaSCI Andalusian Research Institute in Data Science and Computational Intelligence

Table of Contents

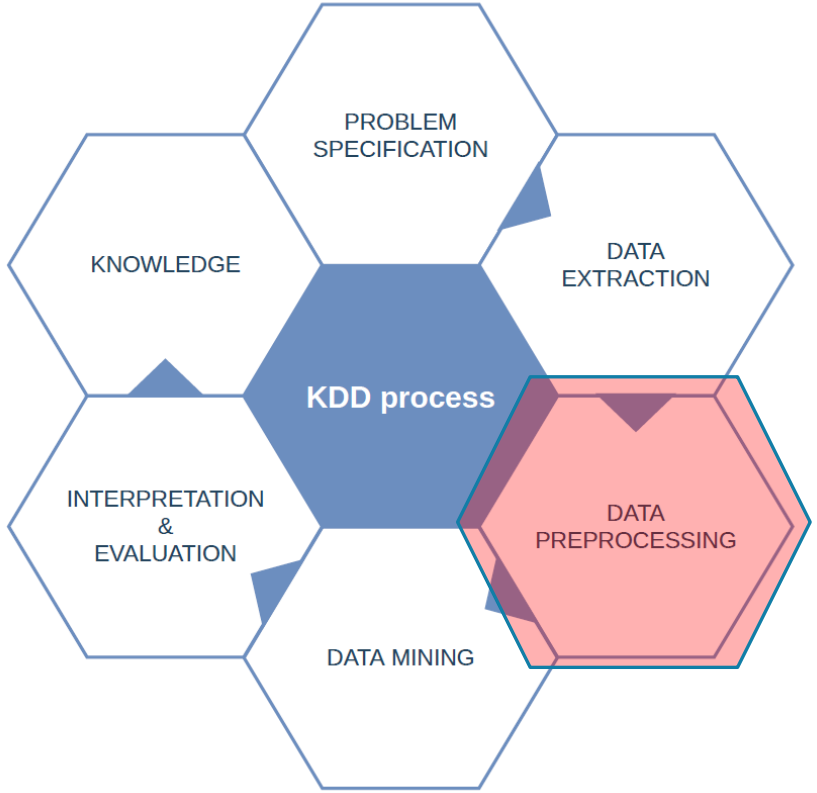




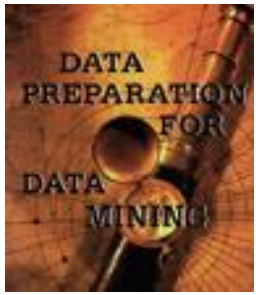
DATA PREPROCESSING | 1

DATA PREPROCESSING

- "Classic concept": Data Mining, KDD
- Data preprocessing: all we do before "generating the model"
  - Data reduction
  - Imperfect data (noise, missing values)
  - Imbalanced data
  - Data integration
  - Etc.

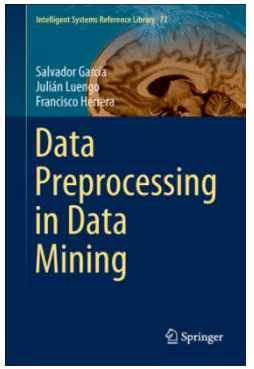


DATA PREPROCESSING



D. Pyle, 1999, pp. 90:

"The fundamental purpose of data preparation is the manipulation and transformation of raw data so that the information contained in the data set can be discovered or made more easily accessible."



S. García, J. Luengo, F. Herrera, 2015, Preface vii:

"Data pre-processing is an often neglected but very important step in the data mining process"

WHY DATA PREPROCESSING (1)

Real data can be impure, can lead to the extraction of unhelpful patterns/rules.

- This can be due to:
  - Incomplete data: missing attribute values, …
  - Data with noise
  - Inconsistent data (including discrepancies)

WHY DATA PREPROCESSING (2)

Data preprocessing can generate a smaller data set than the original, which can improve the efficiency of the Data Mining process.

- This performance includes:
  - Relevant data selection: eliminating duplicate records, eliminating anomalies, ...
  - Data reduction: Characteristics selection, sampling or instance selection, discretization.

WHY DATA PREPROCESSING (3)

Data preprocessing generates "quality data", which can lead to "quality models".

- For example, it can:
  - Retrieve incomplete information.
  - Remove outliers
  - Resolving conflicts
  - Select relevant variables, ...

DATA PREPROCESSING

- Without quality data, there are no quality models
- Data preprocessing techniques generate *quality data* → quality models

Quality decisions must based on quality data

DATA PREPARATION VS. DATA REDUCTION

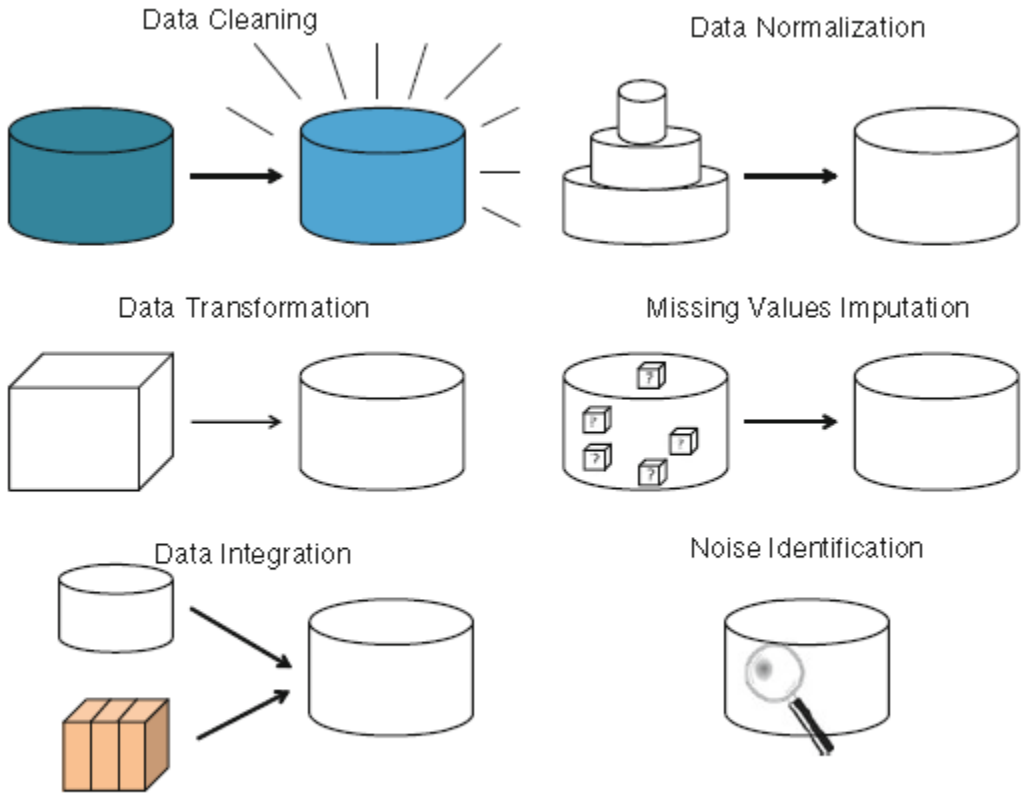


Fig. 1.3 Forms of data preparation

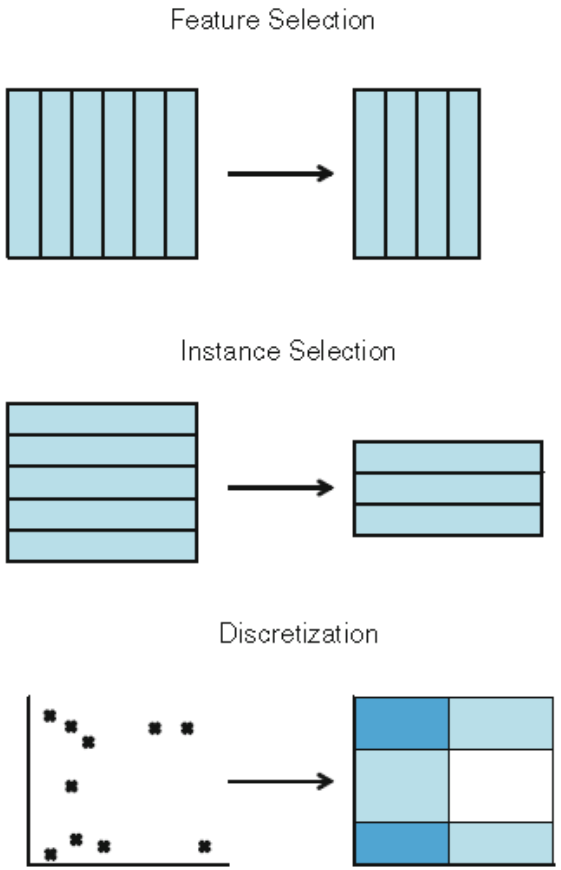
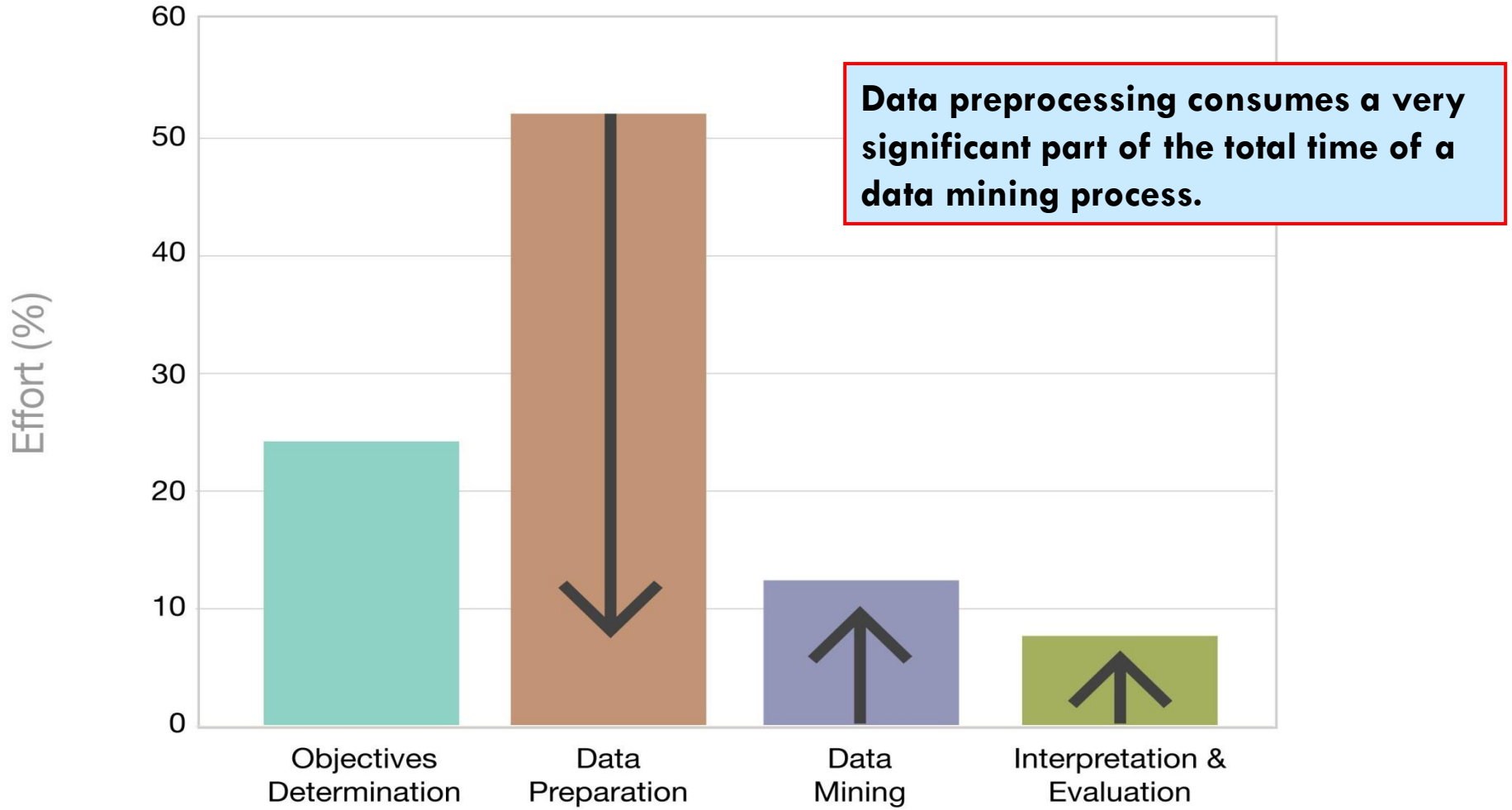
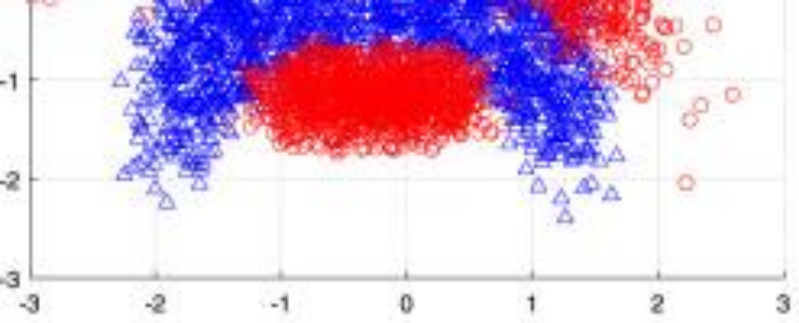


Fig. 1.4 Forms of data reduction

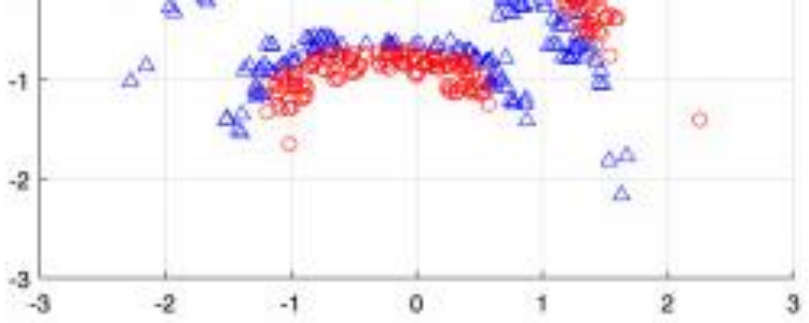
PREPROCESAMIENTO DE DATOS



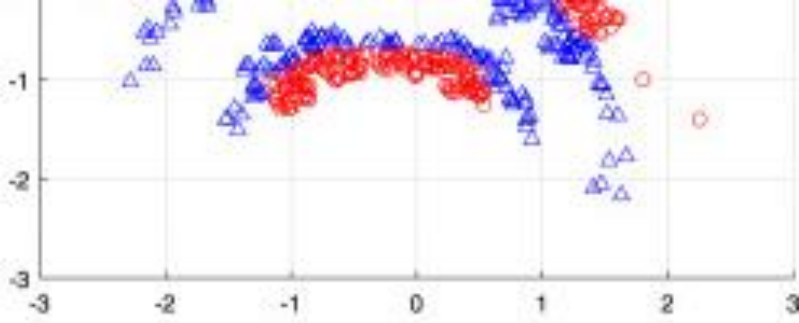
Data preprocessing consumes a very significant part of the total time of a data mining process.



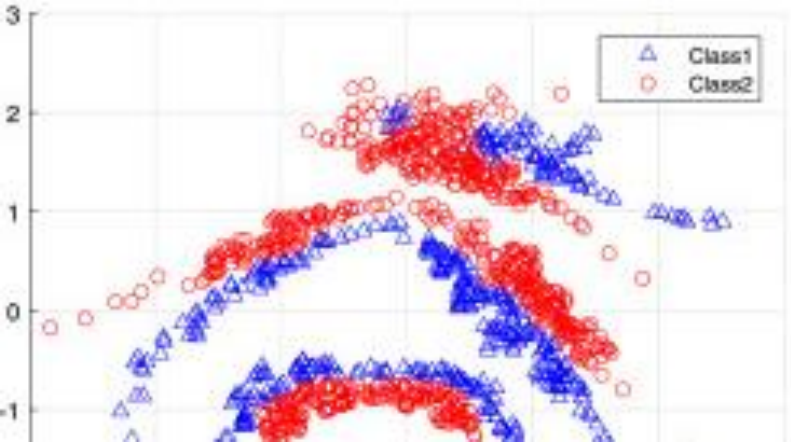
(a)

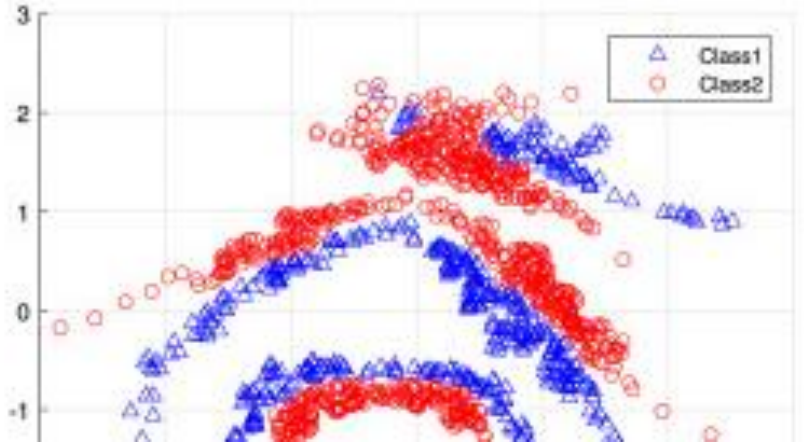


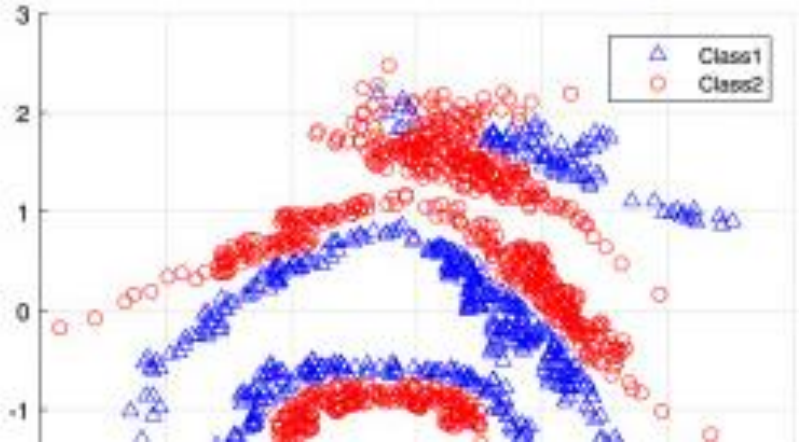
(b) $K = 3$, storage = 15.06%



(c) $K = 5$, storage = 19.30%

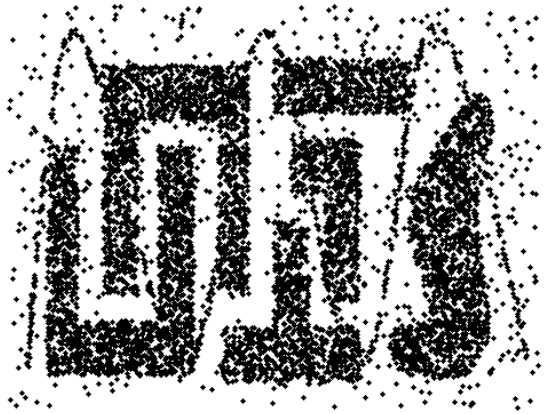




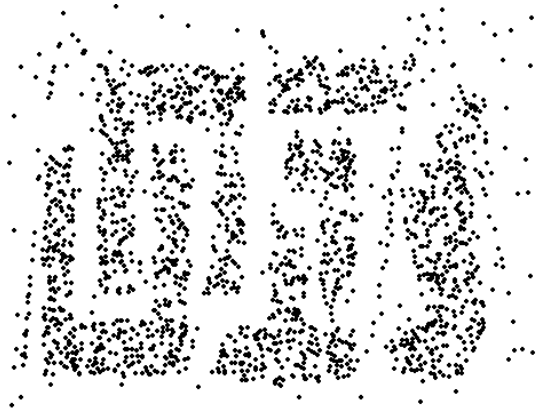


INSTANCE SELECTION | 2

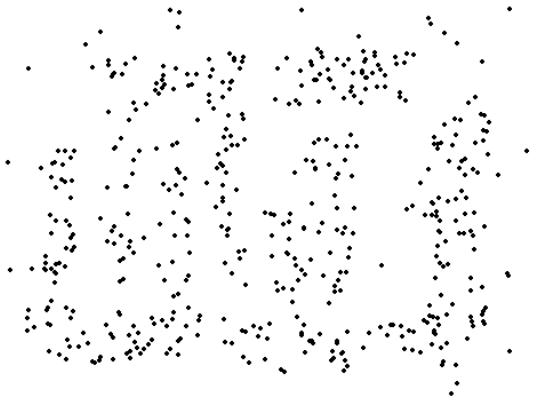
INSTANCE SELECTION



8000 examples



2000 examples



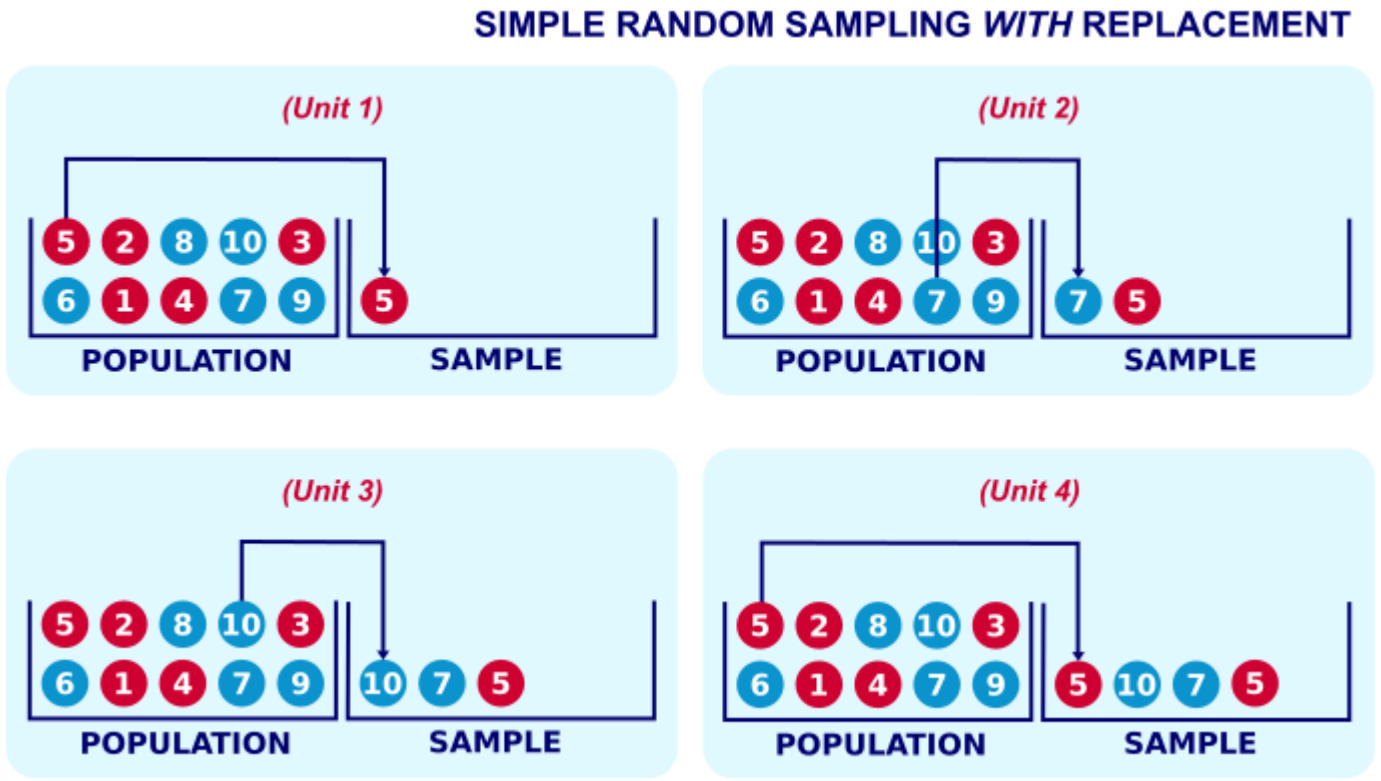
500 examples

INSTANCE SELECTION

- Instance selection aims to choose the examples that are relevant to an application and to achieve maximum performance. The result would be:
  - Less data → algorithms can "learn" faster
  - More accuracy → the model generalizes better
  - Simpler results → easier to understand

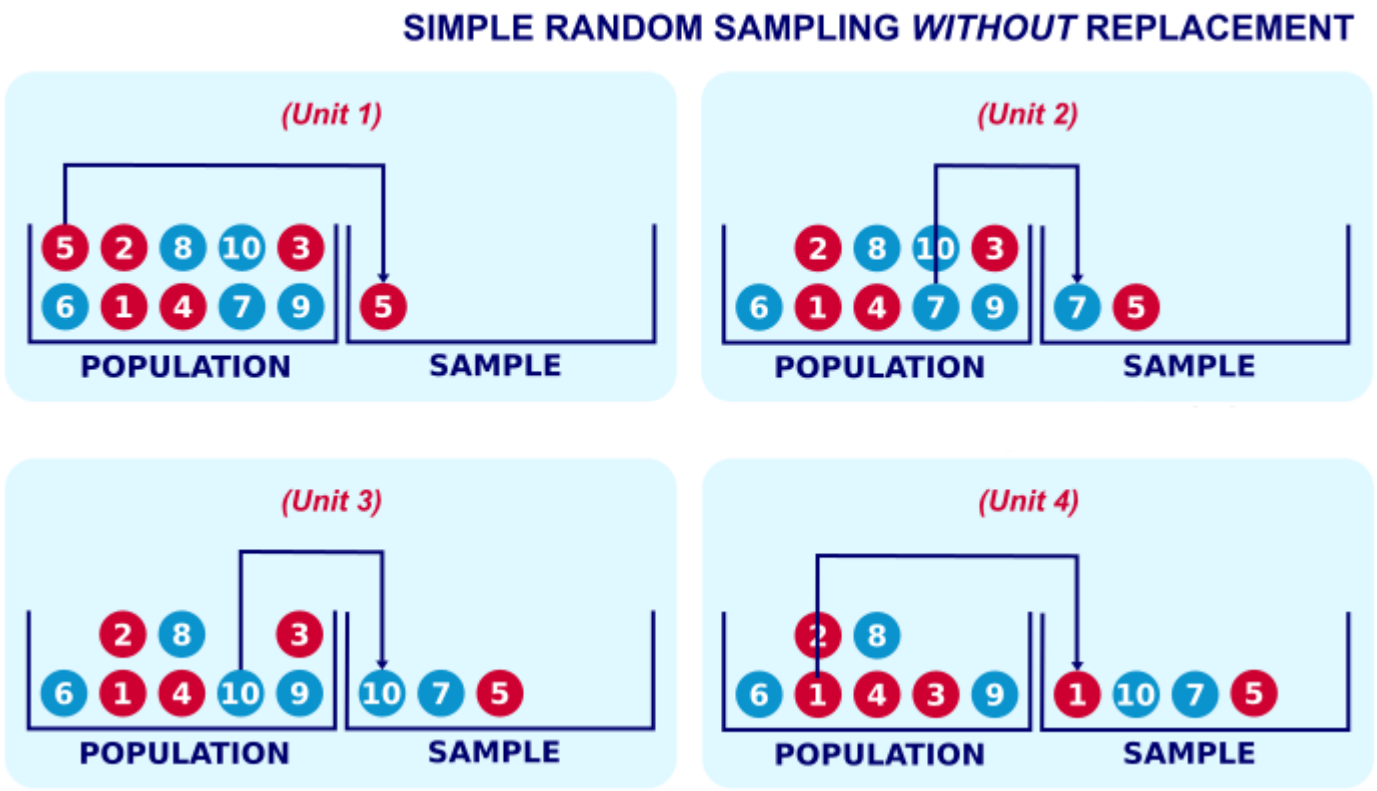
- Supervised vs. unsupervised

SIMPLE RANDOM SAMPLING

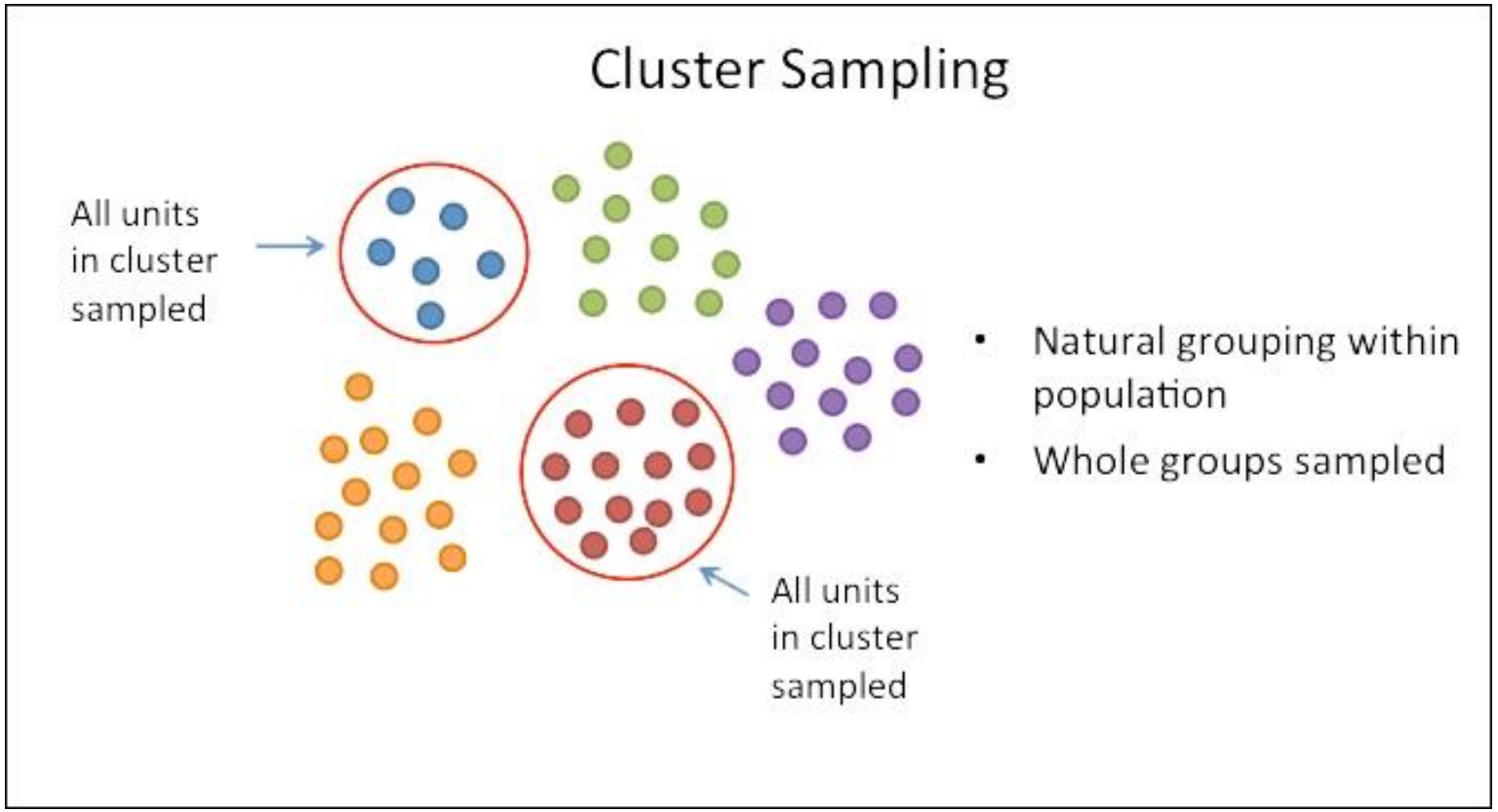


Each time we sample a unit, all units have similar chances of being sampled.

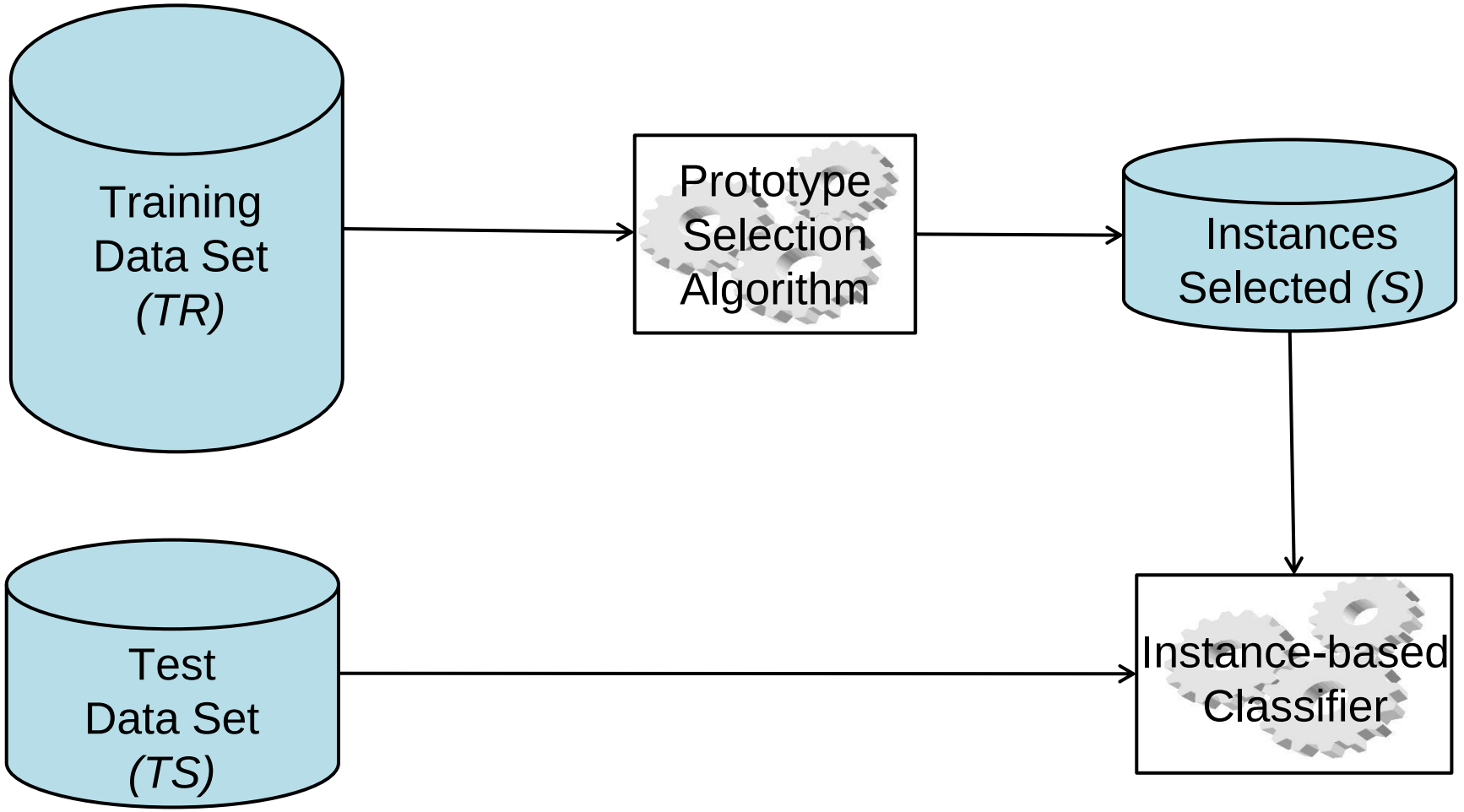
SIMPLE RANDOM SAMPLING



CLUSTER SAMPLING (UNSUPERVISED)



INSTANCE (PROTOTYPE) SELECTION



PROPERTIES OF PS

- Search direction: Incremental, decremental, batch, mixed and fixed
- Type of selection: Condensation, Edition, Hybrid.
- Type of evaluation: Filter or wrapper.

INSTANCE SELECTION

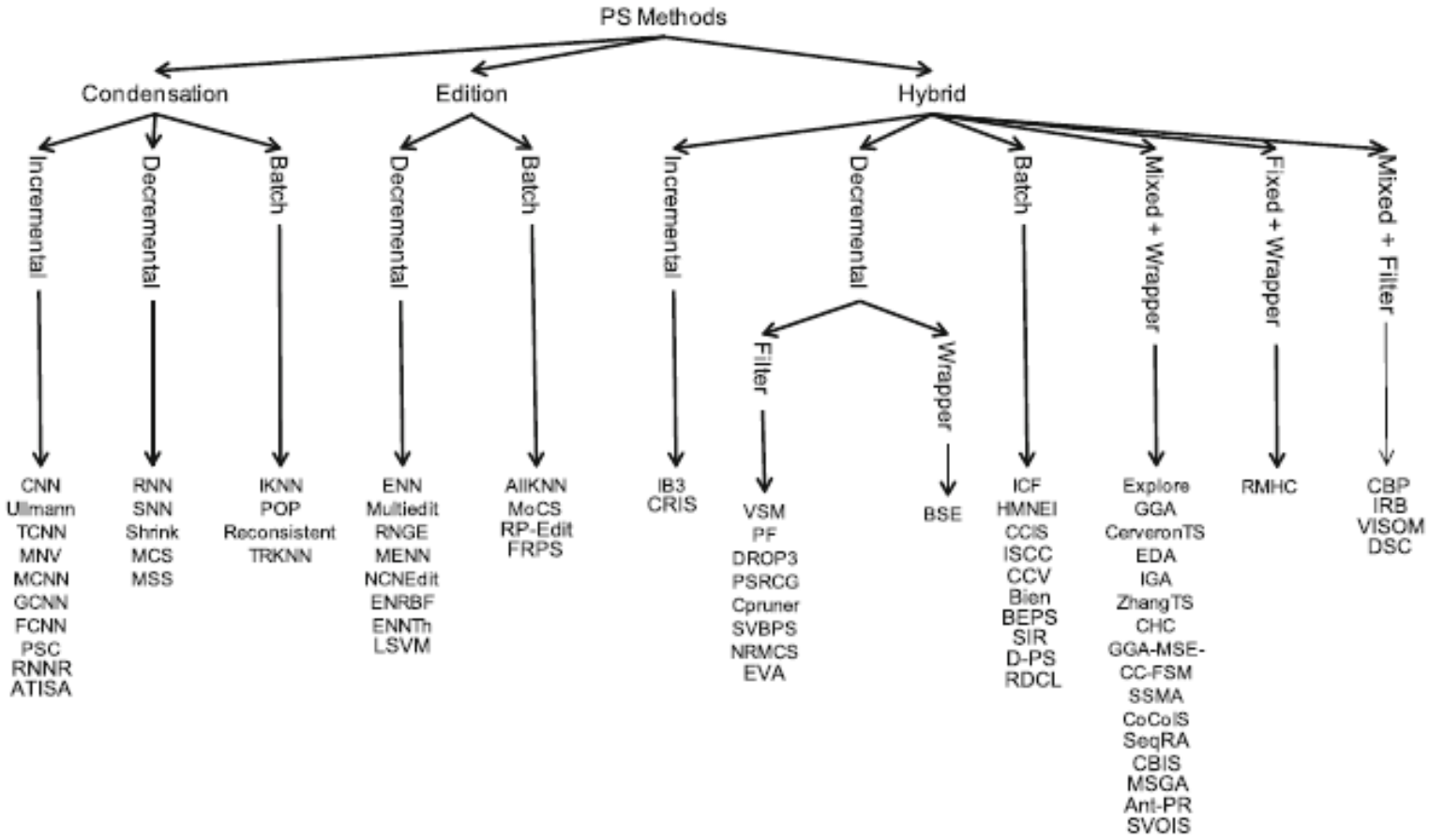
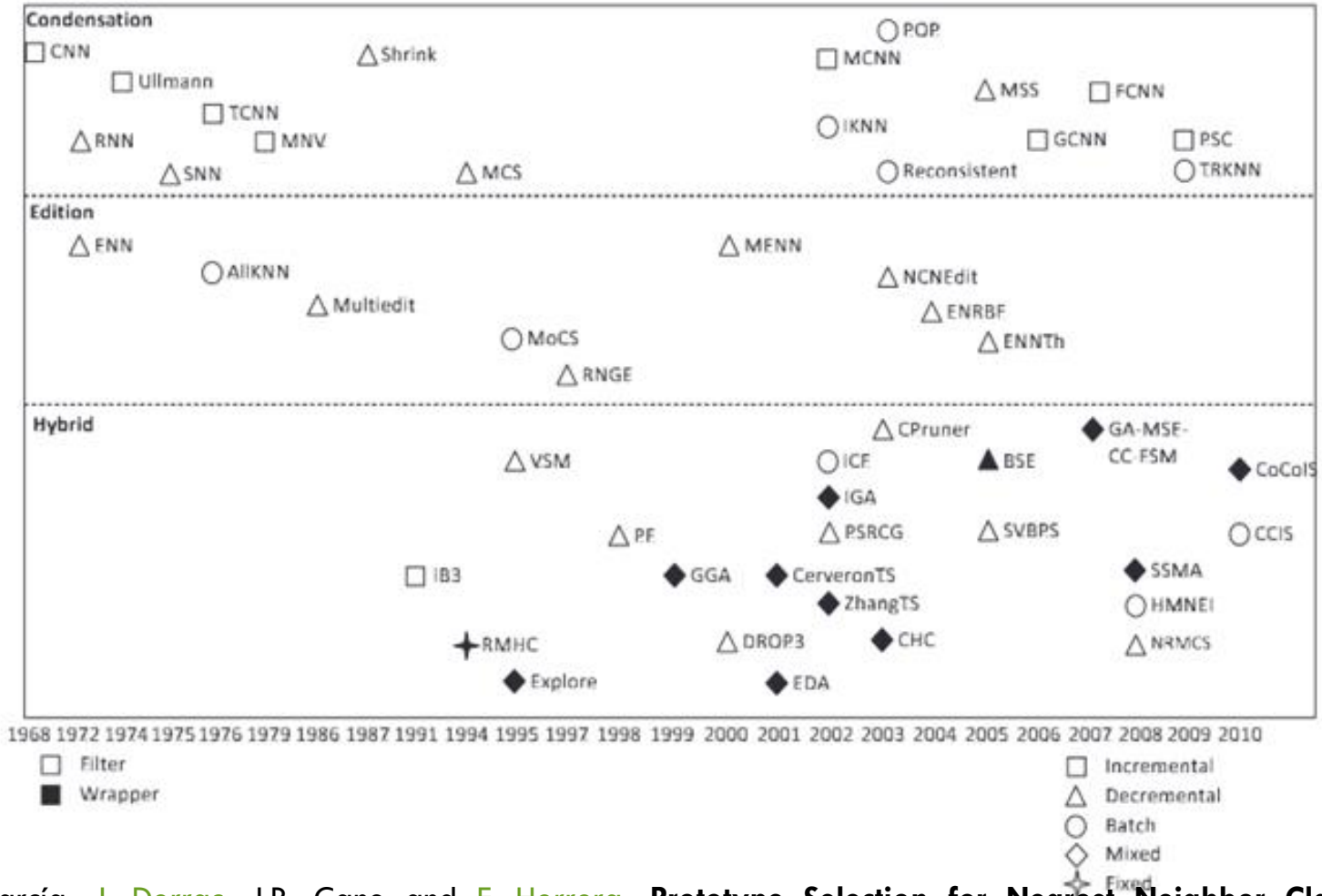


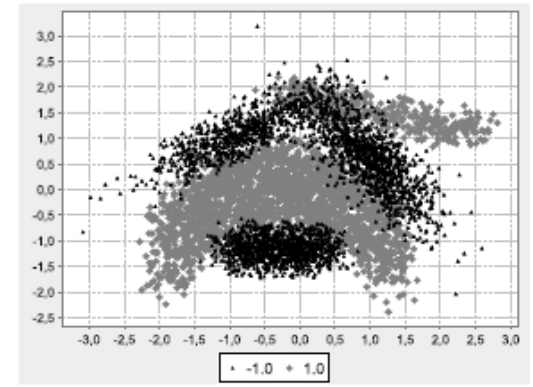
Fig. 8.3 PS taxonomy

INSTANCE SELECTION

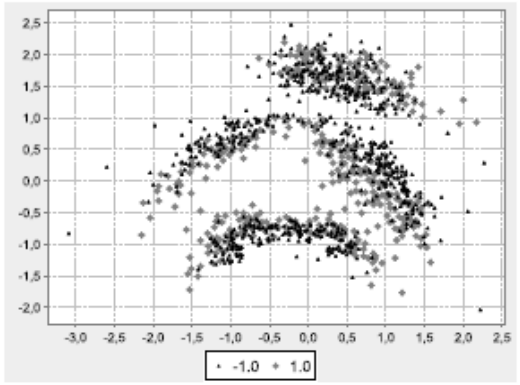


Ref. S. García, J. Derrac, J.R. Cano and F. Herrera, **Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study.** *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34:3 (2012) 417-435 doi: 10.1109/TPAMI.2011.142

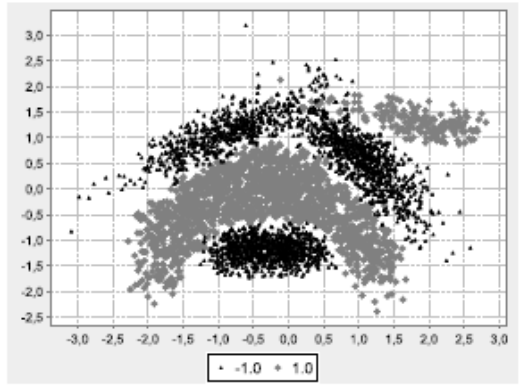
INSTANCE SELECTION EXAMPLES



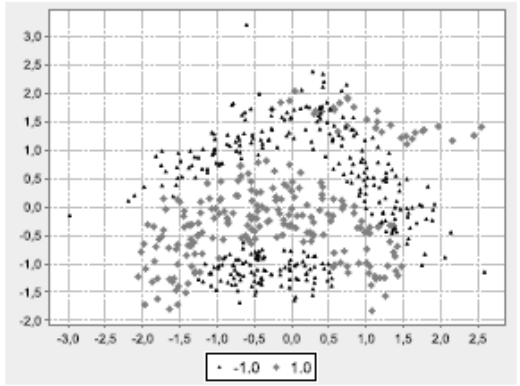
(a) Banana Original (0.8751, 0.7476)



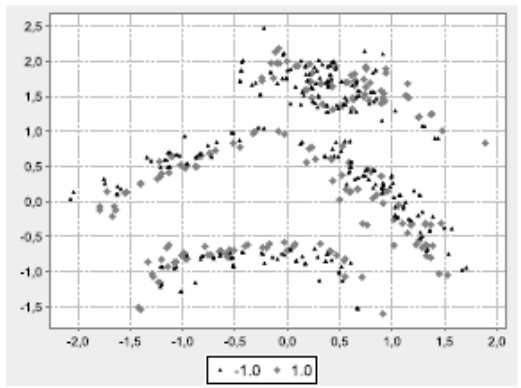
(b) CNN (0.7729, 0.8664, 0.7304)



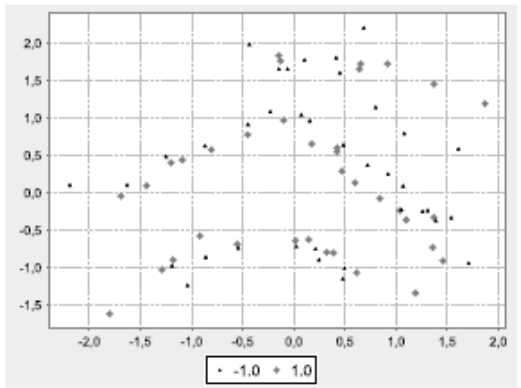
(h) AllKNN (0.1758, 0.8934, 0.7831)



(k) RMHC (0.9000, 0.8972, 0.7915)

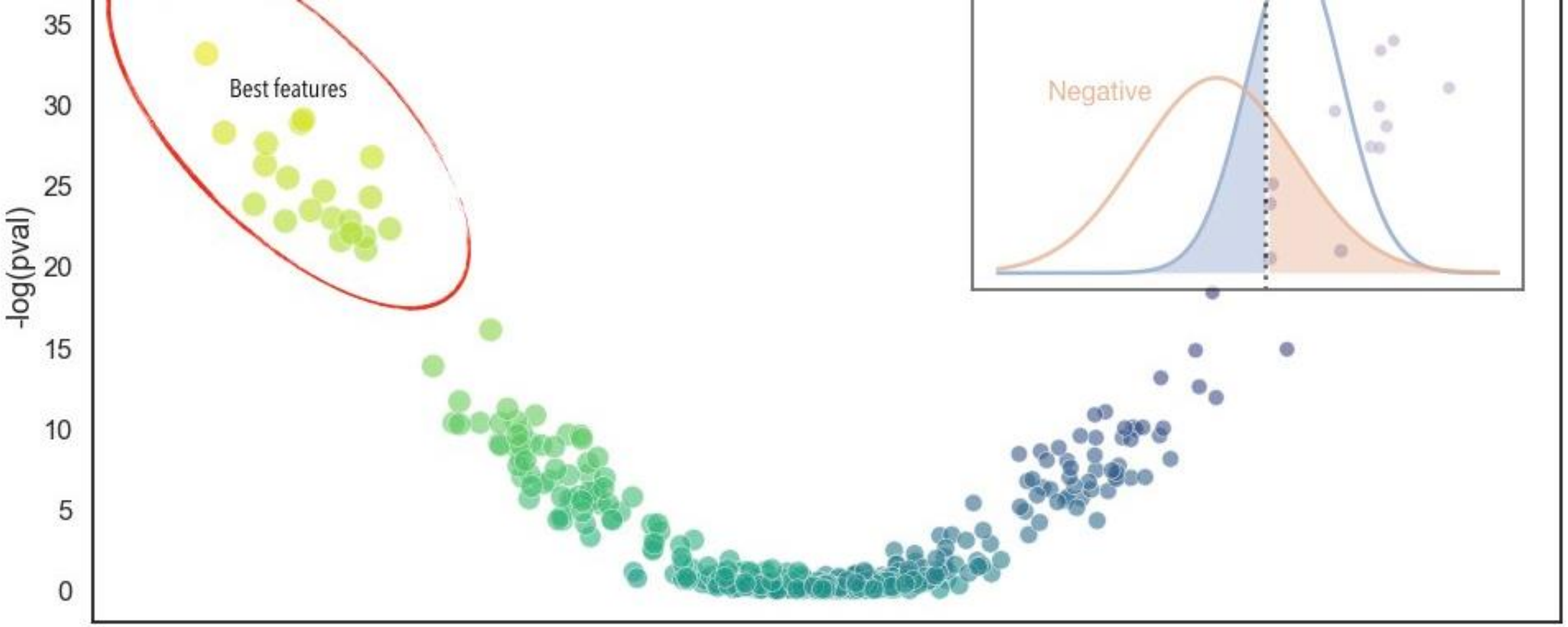


(e) DROP3 (0.9151, 0.8696, 0.7356)



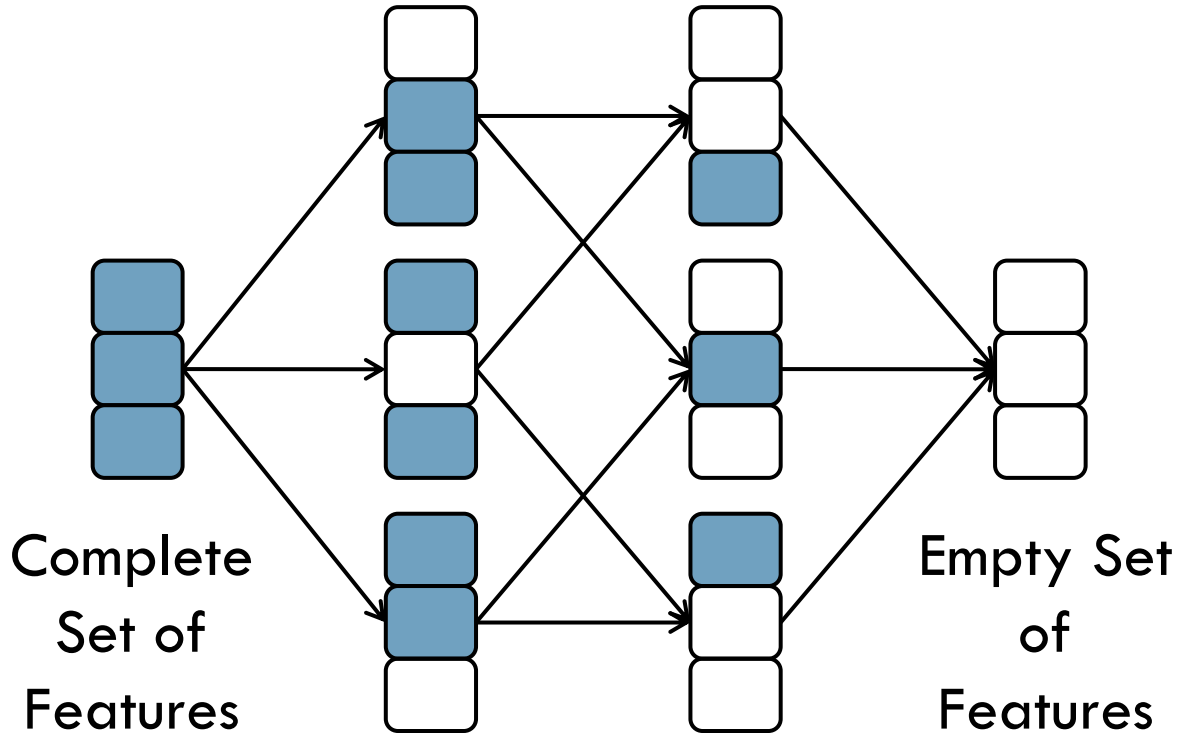
(l) SSMA (0.9879, 0.8964, 0.7900)

Computational cost

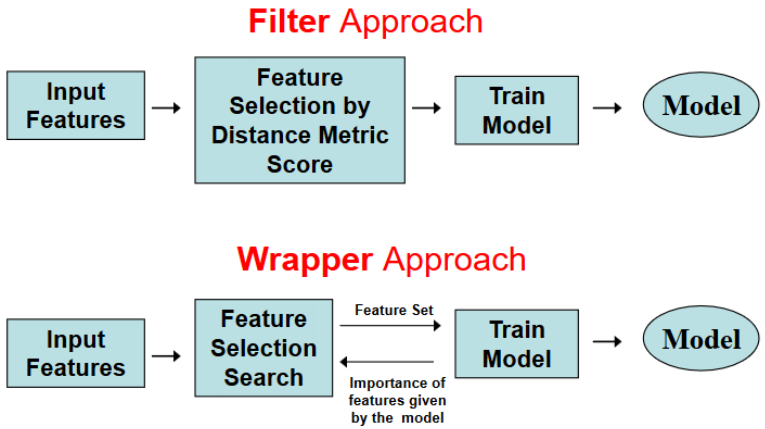


FEATURE SELECTION | 3

FEATURE SELECTION



- NP problem
- Optimal solution is unfeasible even in traditional data sets
- Approximate solutions are also intensive to evaluate



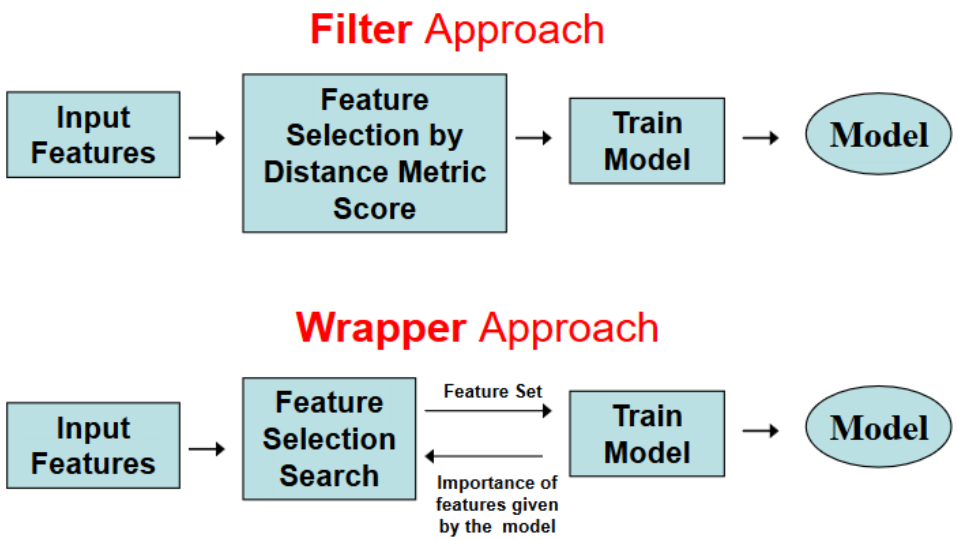
NON SUPERVISED MEASURES

- Used in filter type→ statistical measures (supervised or unsupervised)
- Typical measure: **mutual information**

$$I(A;B) = H(A) - H(A|B)$$
$$= \sum_{a\in A}\sum_{b\in B} p(a|b)\log\frac{p(a|b)}{p(a)p(b)}.$$

- In supervised problems, we examine the MI with respect to the output value
- In unsupervised problems, matrices of MI are analyzed

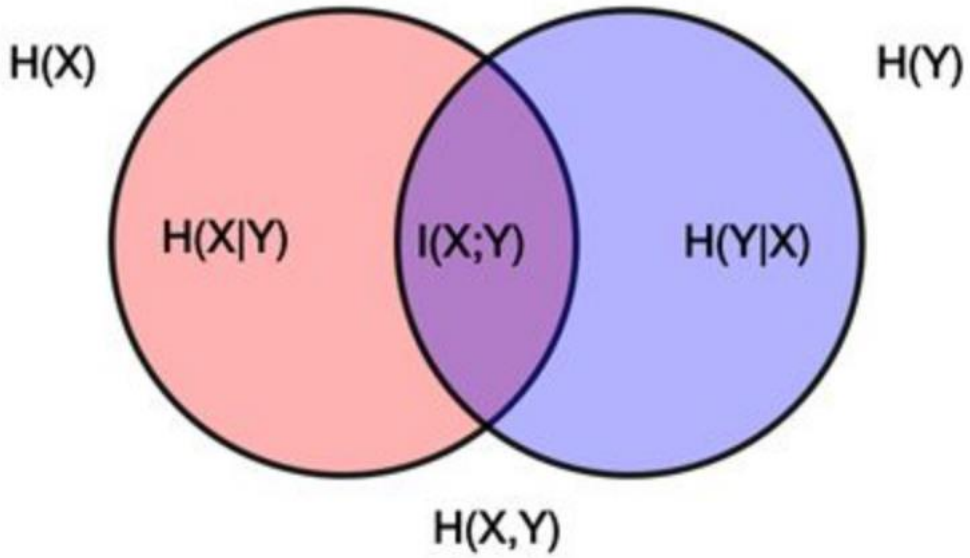
LAS VEGAS FILTER/WRAPPER(LVF/LVW)



- Random generation of subsets of features
- LVF uses the inconsistency rate to evaluate each
- If we use the accuracy of a ML algorithm then we have LVW
  - A 10-fcv is usually used to evaluate the subset of features

H. Liu, R. Setiono, A probabilistic approach to feature selection - a filter solution, in: Proceedings of the International Conference on Machine Learning (ICML), 1996, pp. 319–327.

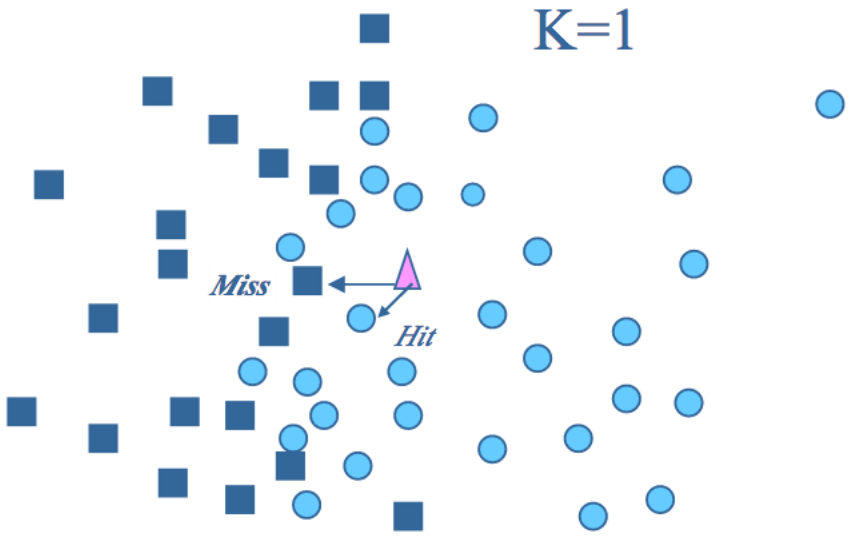
MUTUAL INFORMATION FEATURE SELECTION (MIFS)



- Single computation of the mutual information measure between two features at the same time
- Greedy
- Incremental
- Lots of extensions!

R. Battiti , Using mutual information for selection features in supervised neural net learning, IEEE Trans. Neural Netw. 5 (1994) 537–550

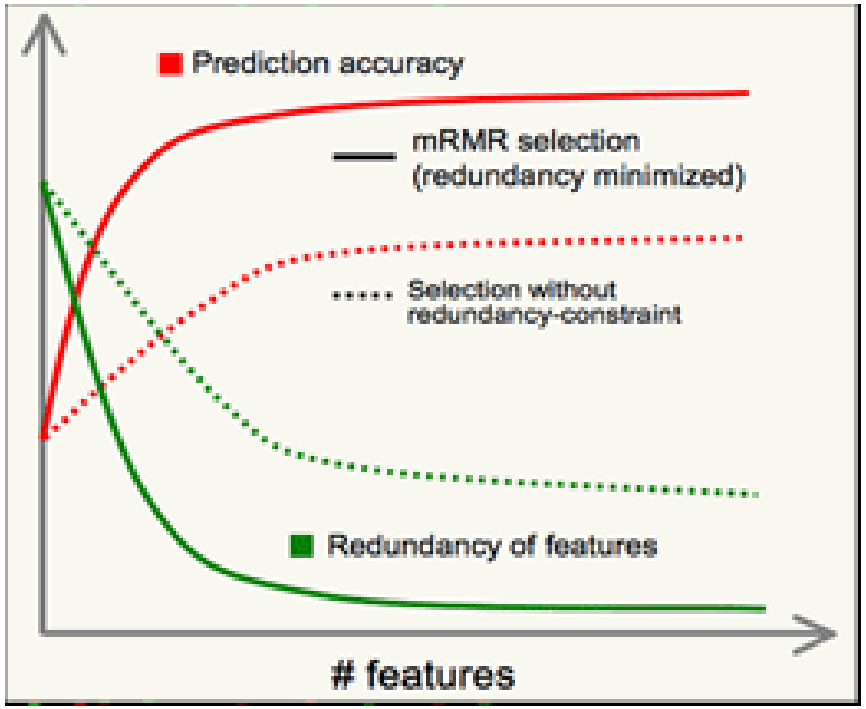
RELIEF



- Selects features according to their statistical relevance.
- Feature weighting.
- Multiclass → ReliefF

K. Kira , L.A. Rendell , A practical approach to feature selection, in: Proceedings of the Ninth International Workshop on Machine Learning, ML92, 1992, pp. 249–256.

MINIMUM REDUNDANCY MAXIMUM RELEVANCE (MRMR)



- It is based on mutual information
- Traditionally, FS → obtain a subset of features with individual high dependency to the target class...
  - ... but the combination of individual good features will not produce a good classification performance
- Redundancy of the selected features can decimate the subset quality
  - Minimizing the redundancy of the features in the selected subset must be also a goal in the search.
- mRMR generates a rank of features.
  - The user must select the amount he/she wants

H. Peng, F. Long, C. Ding, Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1226–1238

COMPARISON

- We apply the FS methods to the *sonar* data set
- Then PCA is used to transform it to 2D
- From these, we can point out the similarities between MIFS and mRMR as they both use the mutual information for their selection process.
- On the other hand, LVF and LVW are not that similar, indicating that the wrapper version behaves quite differently.

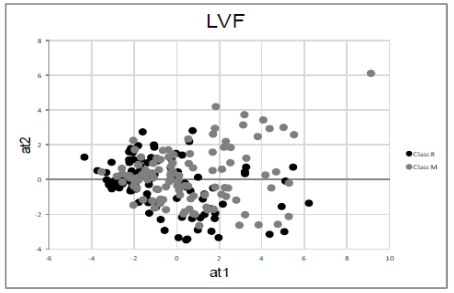


Figure 12: PCA depiction of sonar data set after LVF FS

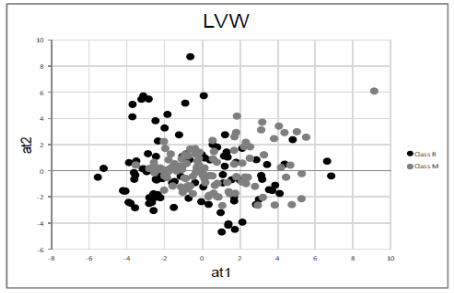


Figure 13: PCA depiction of sonar data set after LVW FS

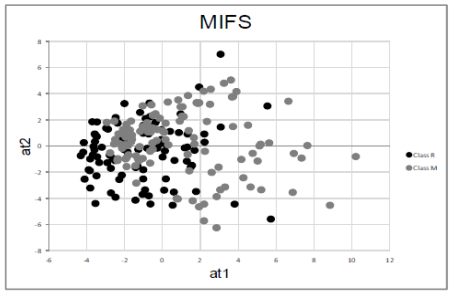


Figure 14: PCA depiction of sonar data set after MIFS FS

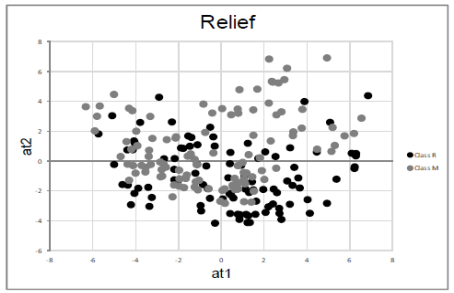


Figure 15: PCA depiction of sonar data set after Relief FS

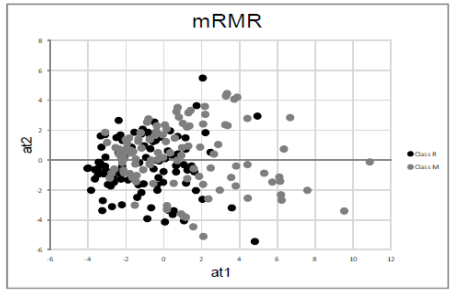
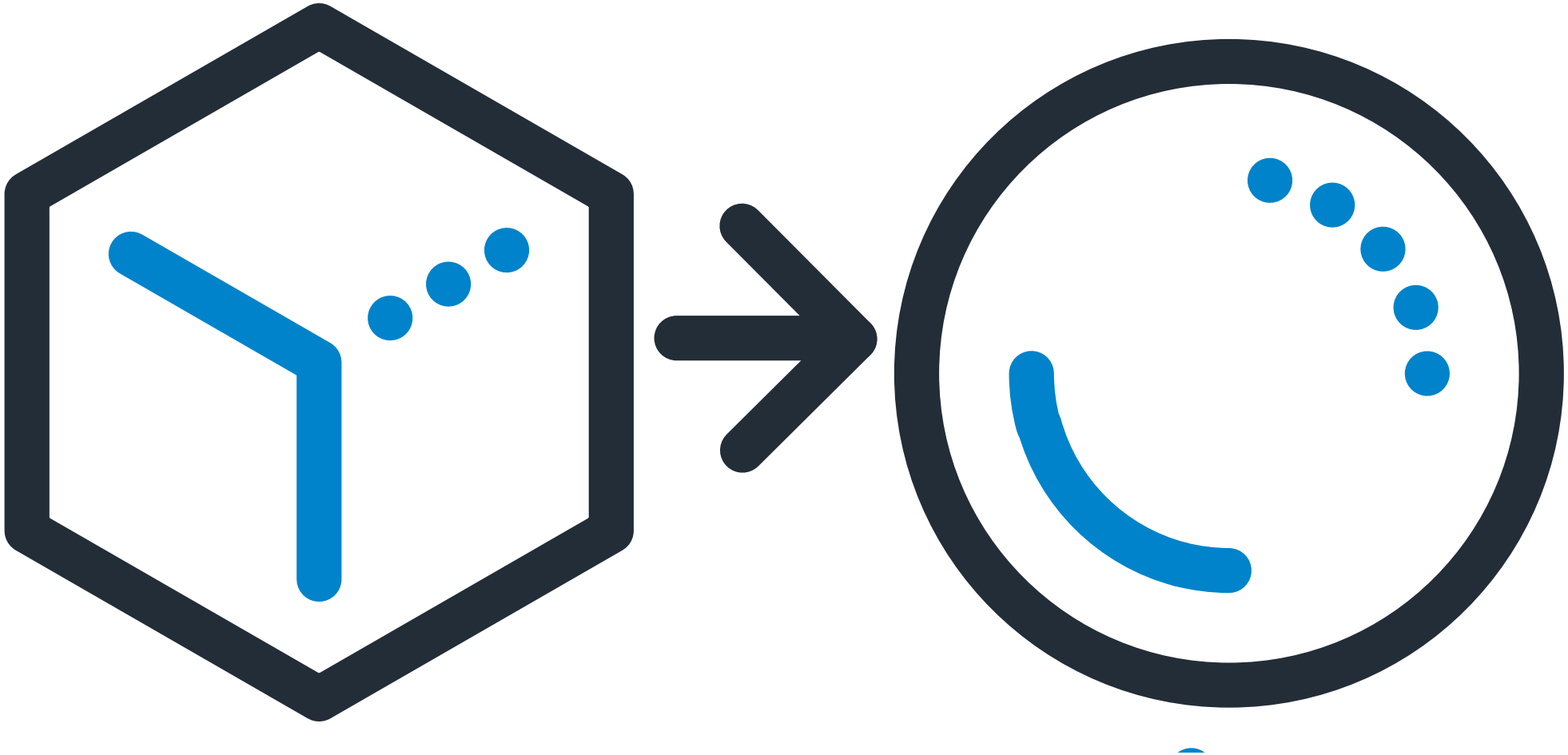
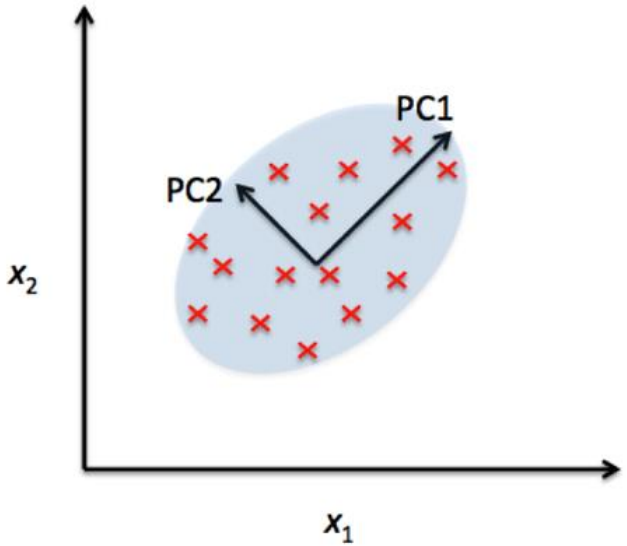


Figure 16: PCA depiction of sonar data set after mRMR FS

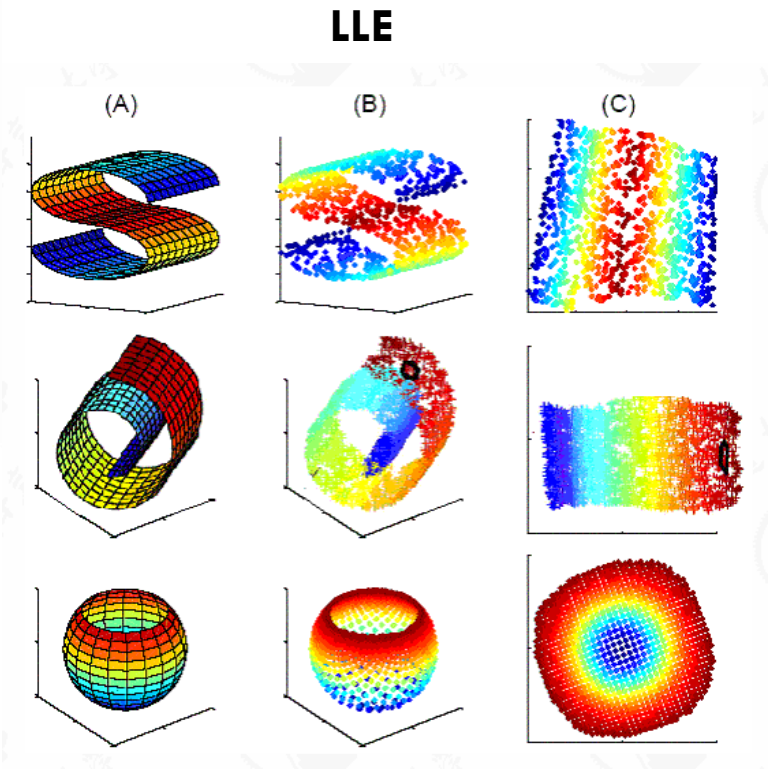


DATA TRANSFORMATION | 4

DATA TRANSFORMATIONS



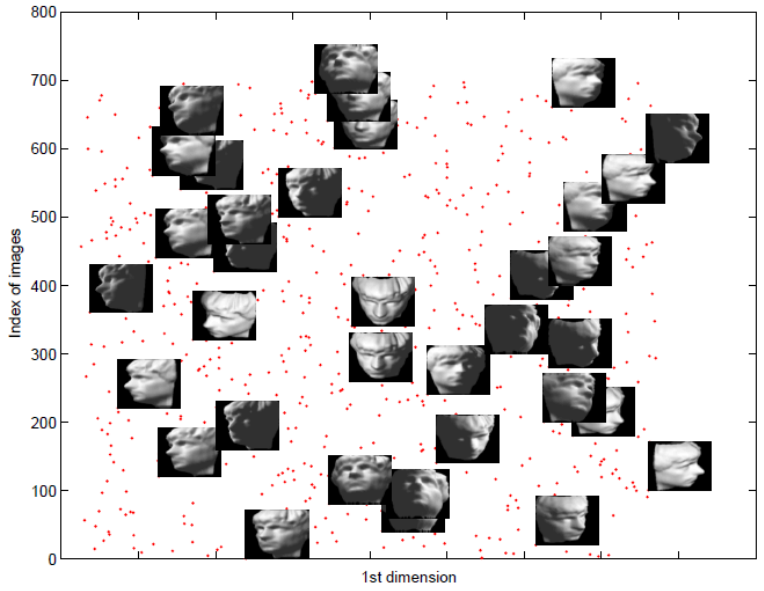
PCA

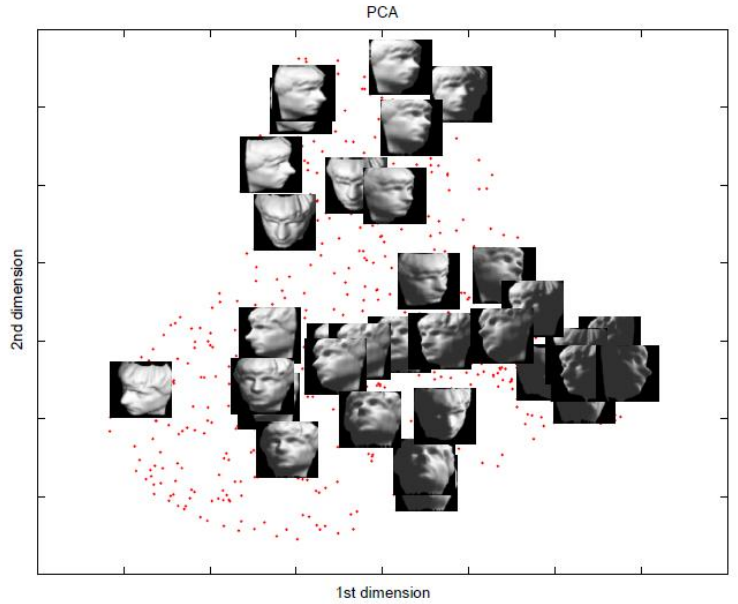


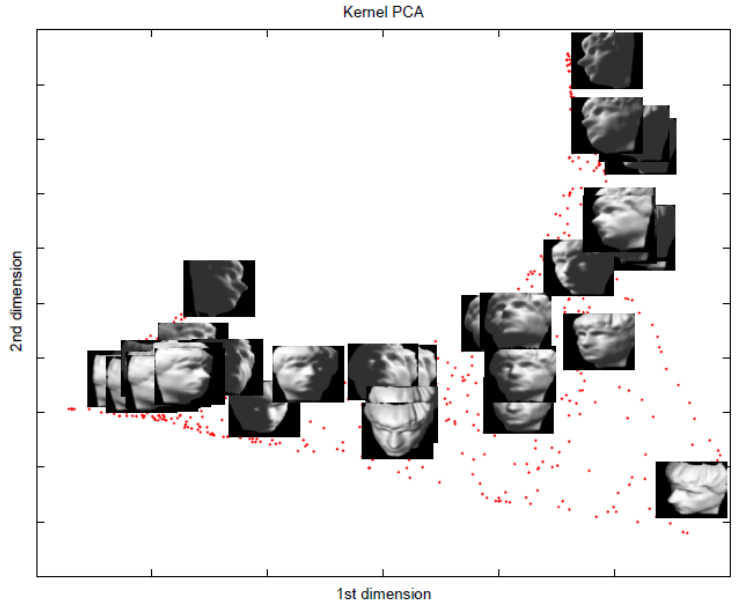
LLE

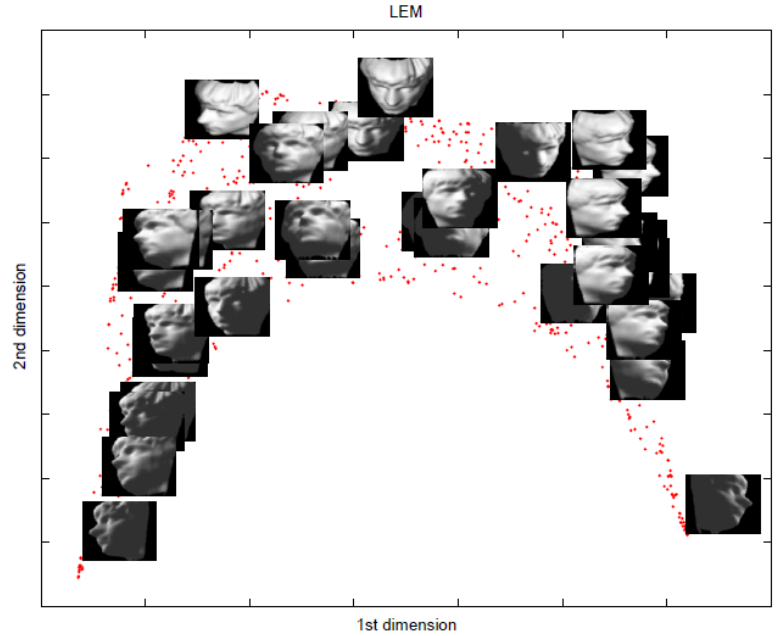
- Changing the input space, instead of proyecting
- Again:
  - Obtaining the optimum is NP
  - Approximate solutions are computationally expensive as well

EXAMPLES


800
700
600
500
400
300
200
100
0
Index of images
1st dimension


PCA
2nd dimension
1st dimension

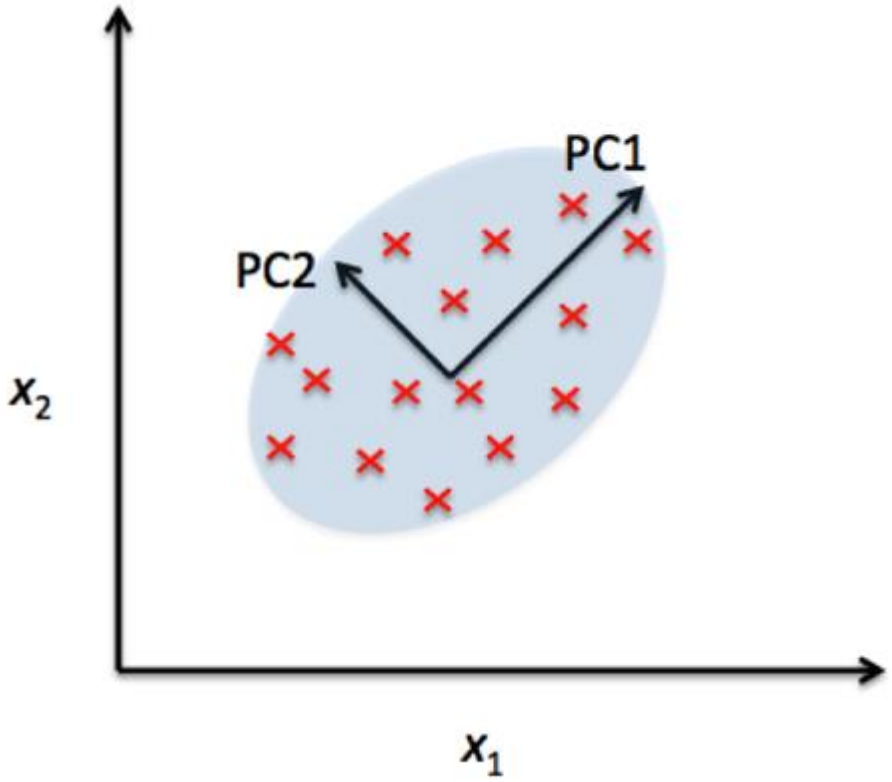

Kernel PCA
2nd dimension
1st dimension


LEM
2nd dimension
1st dimension

LINEAR VS. NON LINEAR

- Linear methods
  - Principal component analysis (PCA)
  - Multidimensional scaling (MDS)
  - Independent component analysis (ICA)
- Non linear methods
  - Kernel PCA
  - Locally linear embedding (LLE)
  - *Laplacian eigenmaps (LEM)*
  - *Semidefinite embedding (SDE)*
  - Autoencoders

PRINCIPAL COMPONENT ANALYSIS (PCA)



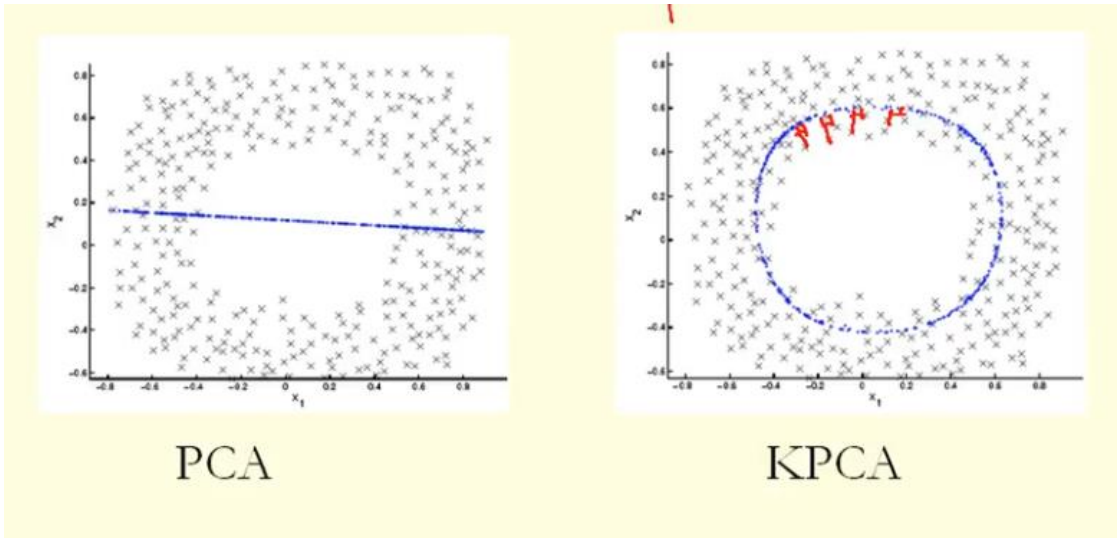
- PCA aims to generate a whole new set of attributes such that they are as independent (or less redundant) as possible.
- PCA creates a set of linear transformations → lower number of variables as a result
- The goal is to maintain only a few principal components that will contain approximately 95% of the variance

I. Jolliffe, Principal component analysis, Wiley Online Library, 2005.

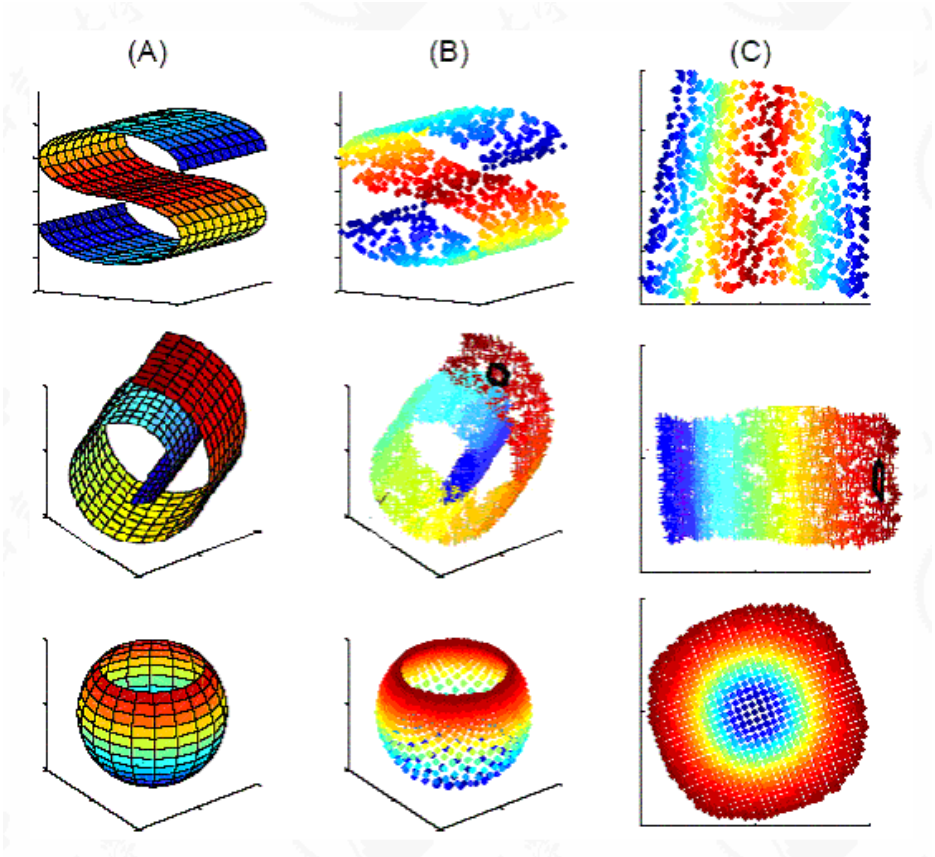
KERNEL PCA

- S. Mika et al, NIPS, 1999
- *Data may lie on or near a nonlinear manifold, not a linear subspace*
- Optimize $\Phi : x \to \mathcal{H} \qquad x \mapsto \Phi(x)$

$$\min_{U_k} \sum_{i=1}^{N} \left\| \Phi(\mathbf{x}_i) - U_k U_k^T \Phi(\mathbf{x}_i) \right\|^2$$



LOCALLY LINEAR EMBEDDING (LLE)



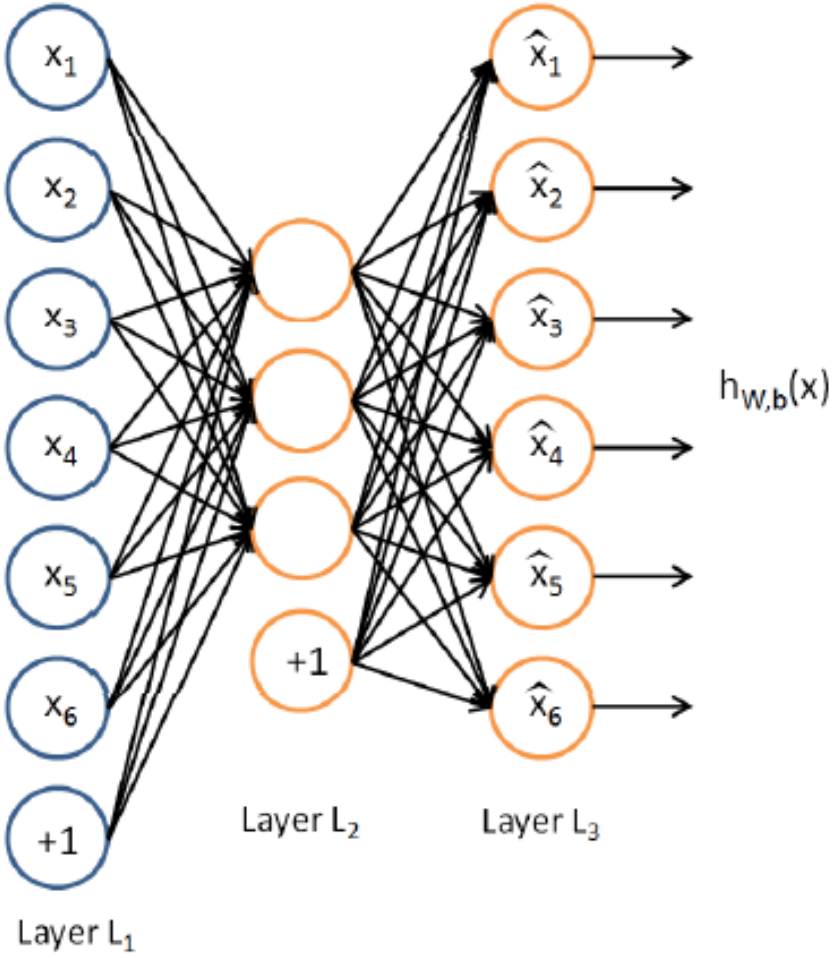
- LLE focuses on the manifolds formed by the data samples
  - Given a manifold, LLE assumes that **a local region of such a manifold can be linearly approximated**.
- An example (a point) can be obtained by a weighted linear combination of its neighbors.
- The grouping of these linear fits serves LLE to recover a complex, nonlinear structure represented in the data.

S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.

AUTOENCODERS

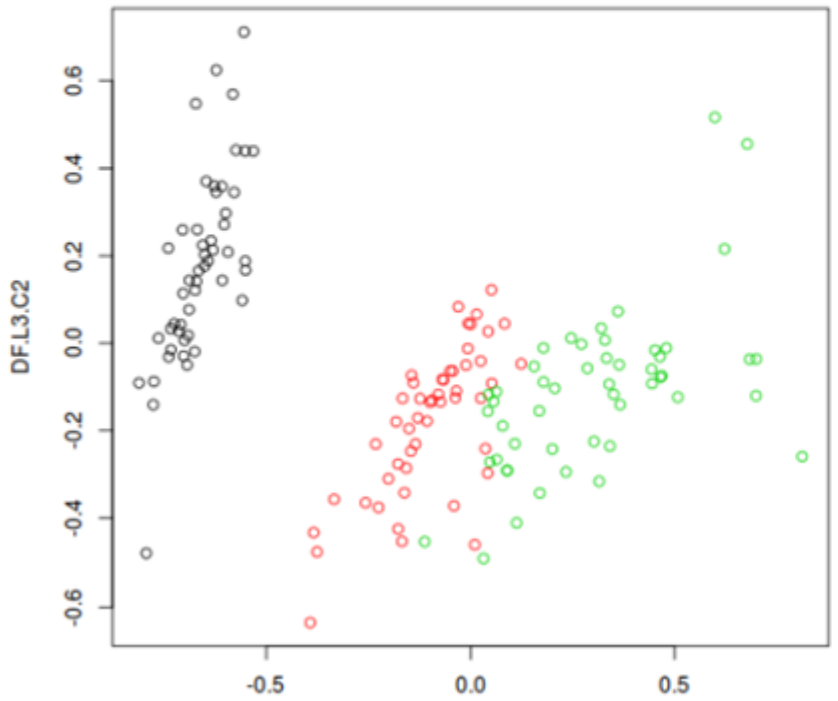
An **autoencoder** neural network is an unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the inputs.

The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction.

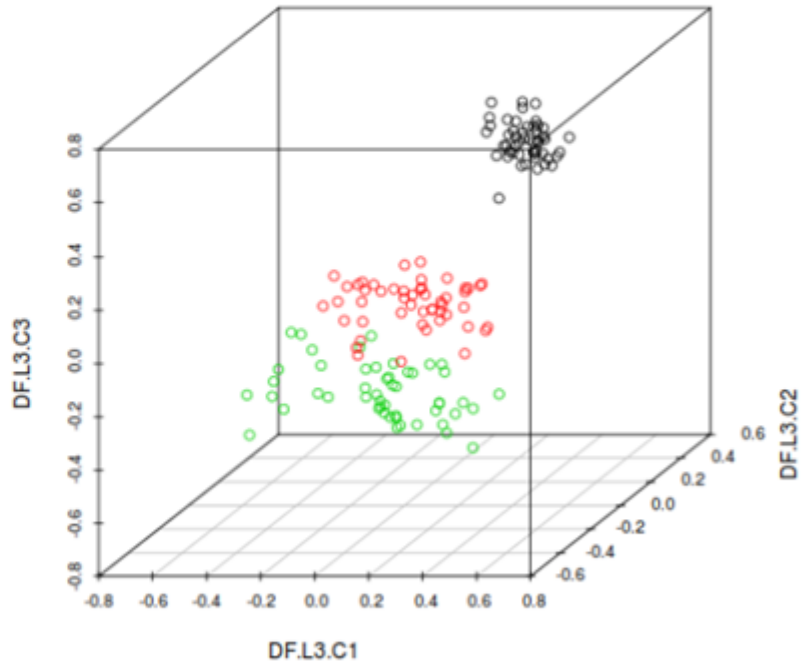


AUTOENCODERS

setosa, versicolor* y *virginica



[8, *5*, **2**, *5*, 8]



[8, *5*, **3**, *5*, 8]

COMPARISON AMONG THE SPACE TRANSFORMATION METHOD

- We apply the space transformation methods to the *sonar* data set
- The transformations are pretty different:
  - PCA aims to maintain the variance of the data
  - LLE tries to conserve the local structures conformed by the manifolds of reduced clusters of examples
- LLE usually arranges the transformed data in recognizable structures, while PCA does not…
- …but LLE tends to be very sensitive to the number of neighbors to build the manifold. PCA converges faster.

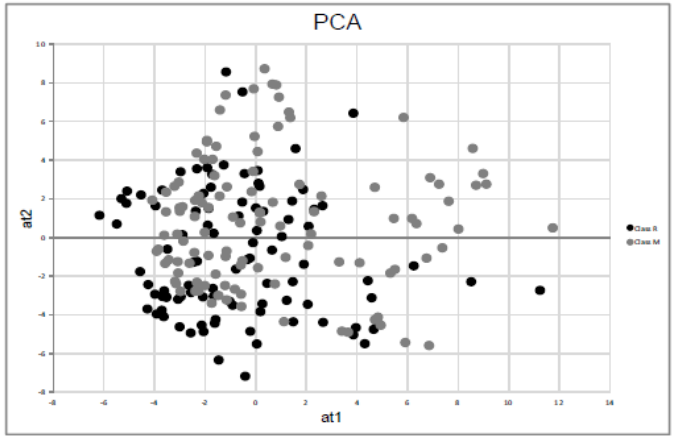


Figure 17: Sonar dimensionality reduced to 2 attributes with PCA

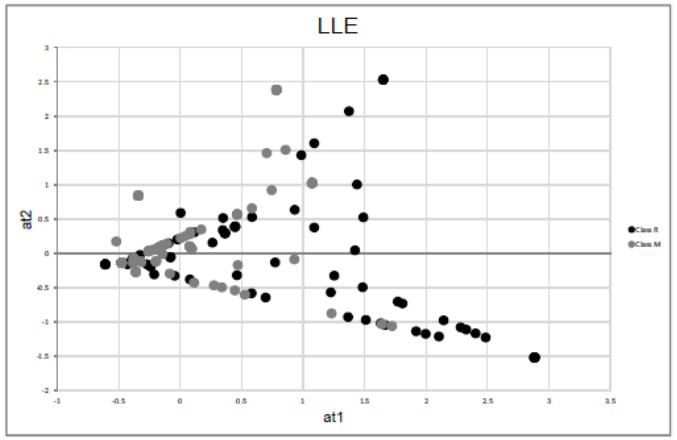
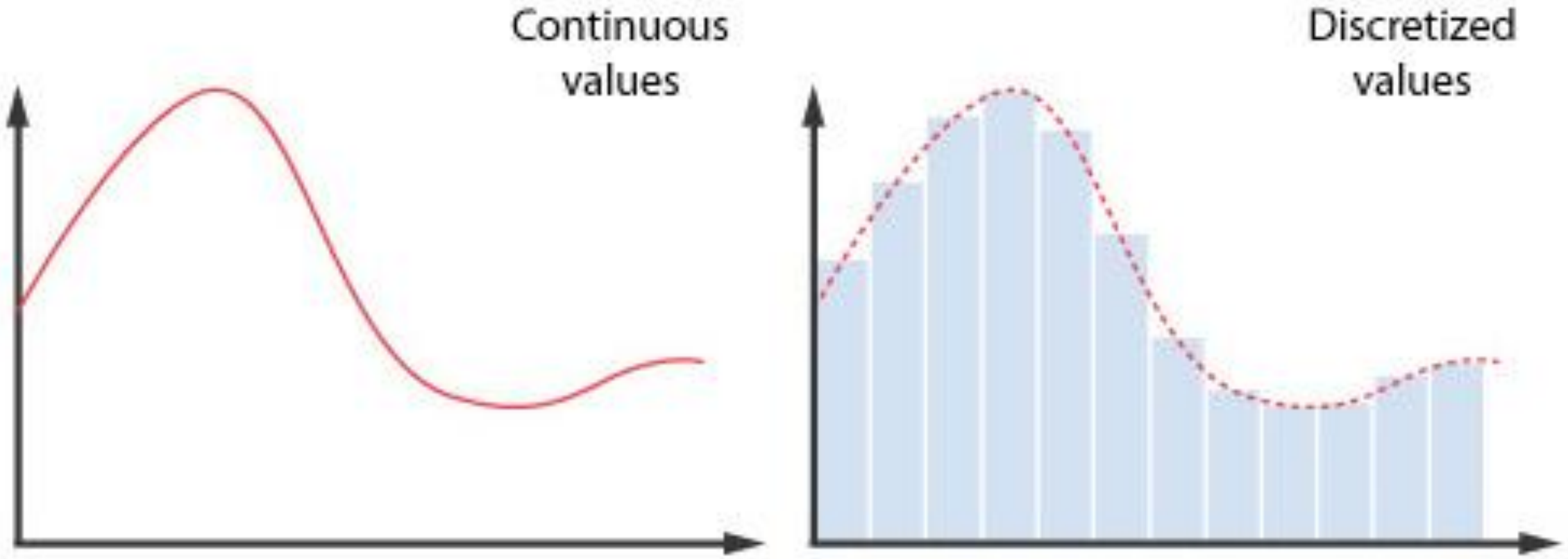


Figure 18: Sonar dimensionality reduced to 2 attributes with LLE

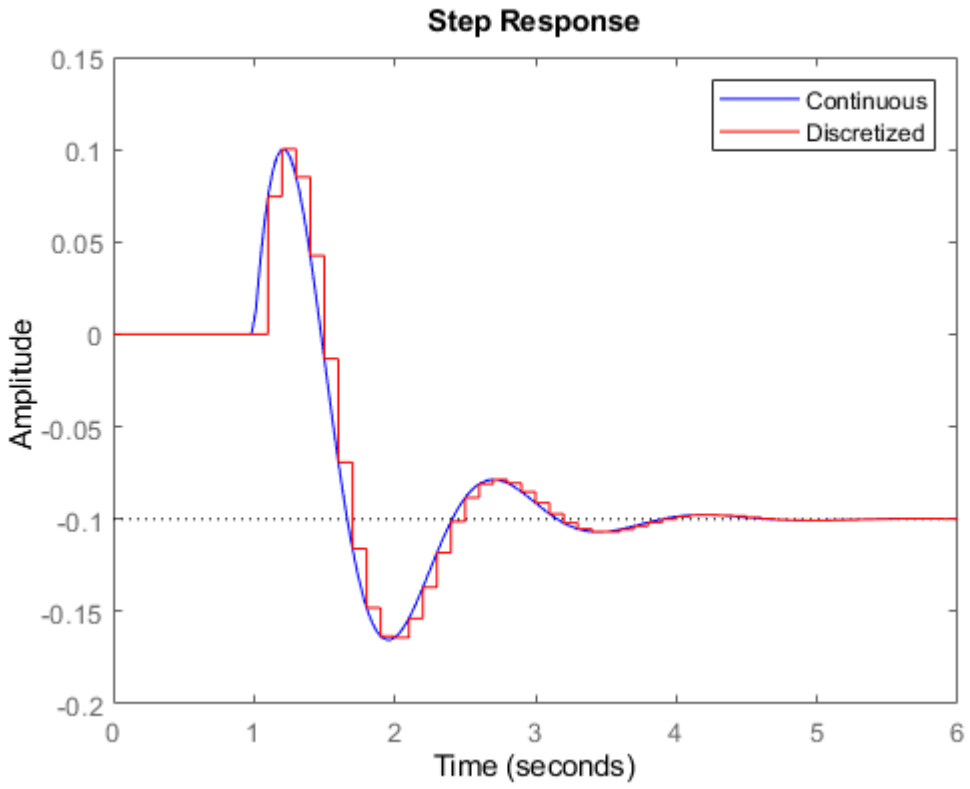


DISCRETIZATION | 5

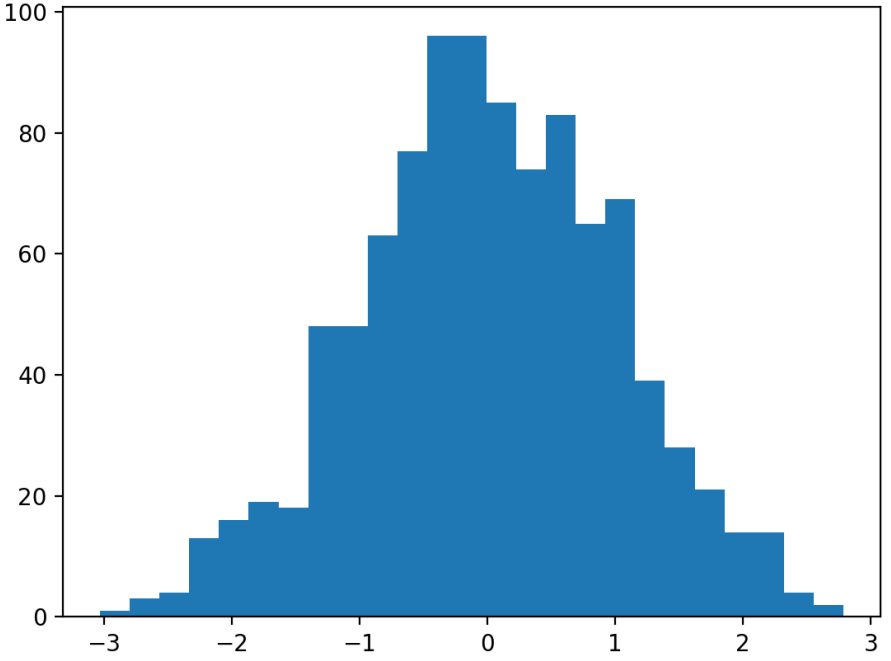
DISCRETIZATION

- Discretization seeks to transform continuous/discrete values that are ordered into nominal values that are not ordered. Process of quantification of numerical attributes.
- They represent more concise information, are easier to understand and closer to knowledge level representation.
- Nominal values have a finite domain, so it is also considered a technique of **data reduction.**
- Discretization can be done before the knowledge gain or during the knowledge extraction stage.

SAMPLING VS. **DISCRETIZATION**



Time is involved

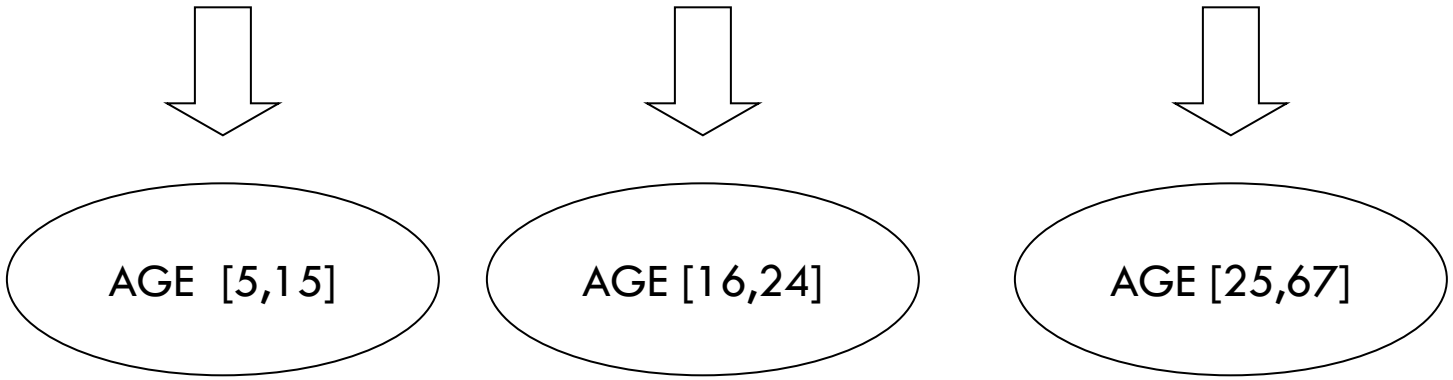


Static data

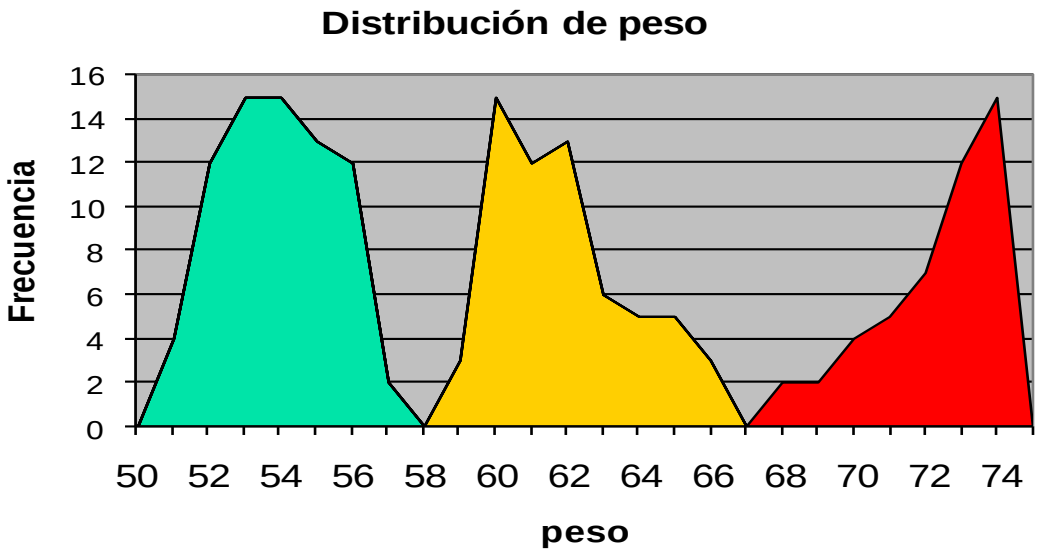
DISCRETIZATION

- In static data we divide the numeric range into intervals
- Intervals are labeled into nominal values

AGE	5	6	6	9	…	15	16	16	17	20	…	24	25	41	50	65	…	67
OWNED CAR	0	0	0	0	…	0	1	0	1	1	…	0	1	1	1	1	…	1



DISCRETIZATION

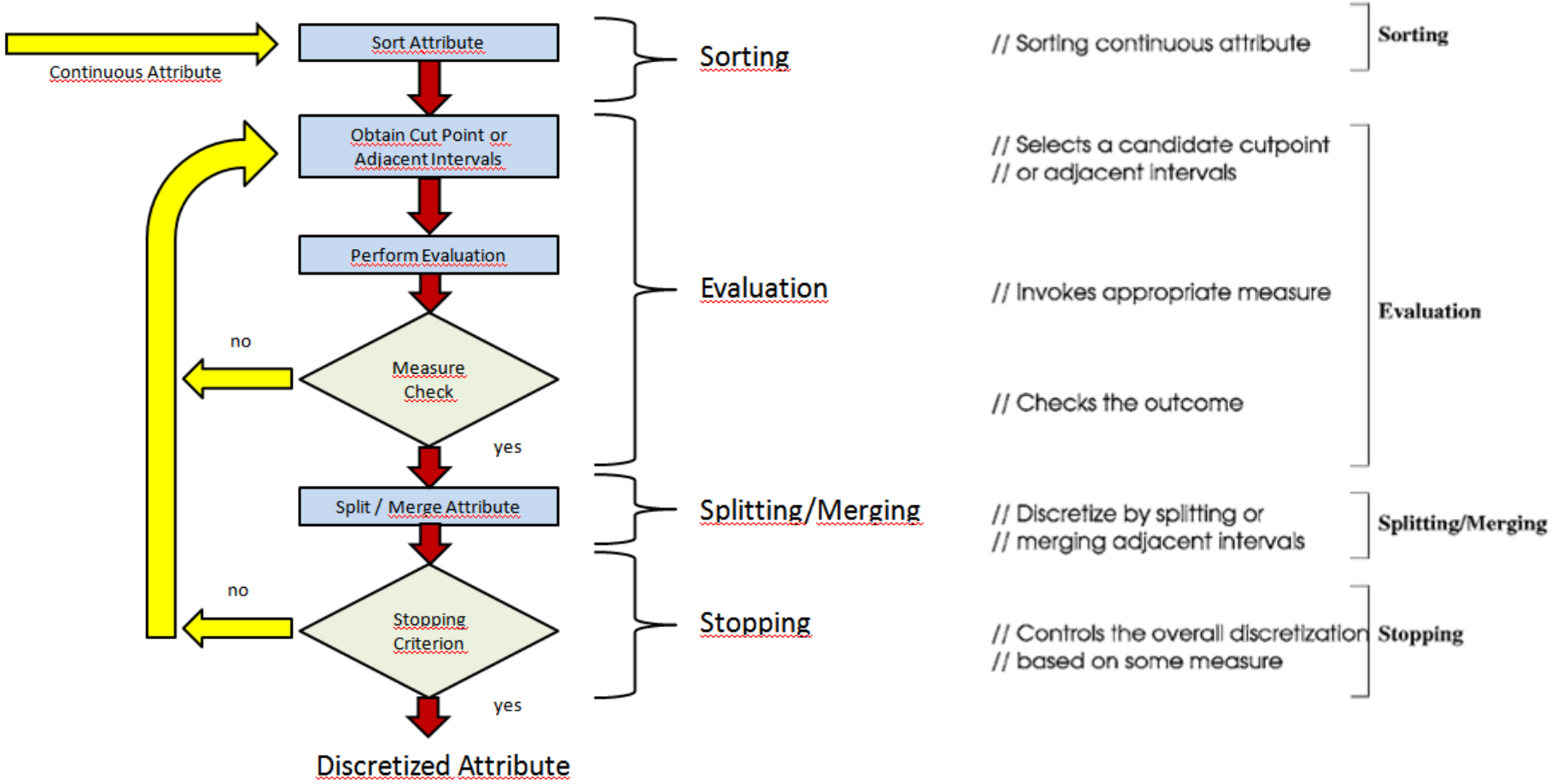


50 - 58 kg

59-67 kg

> 68 kg

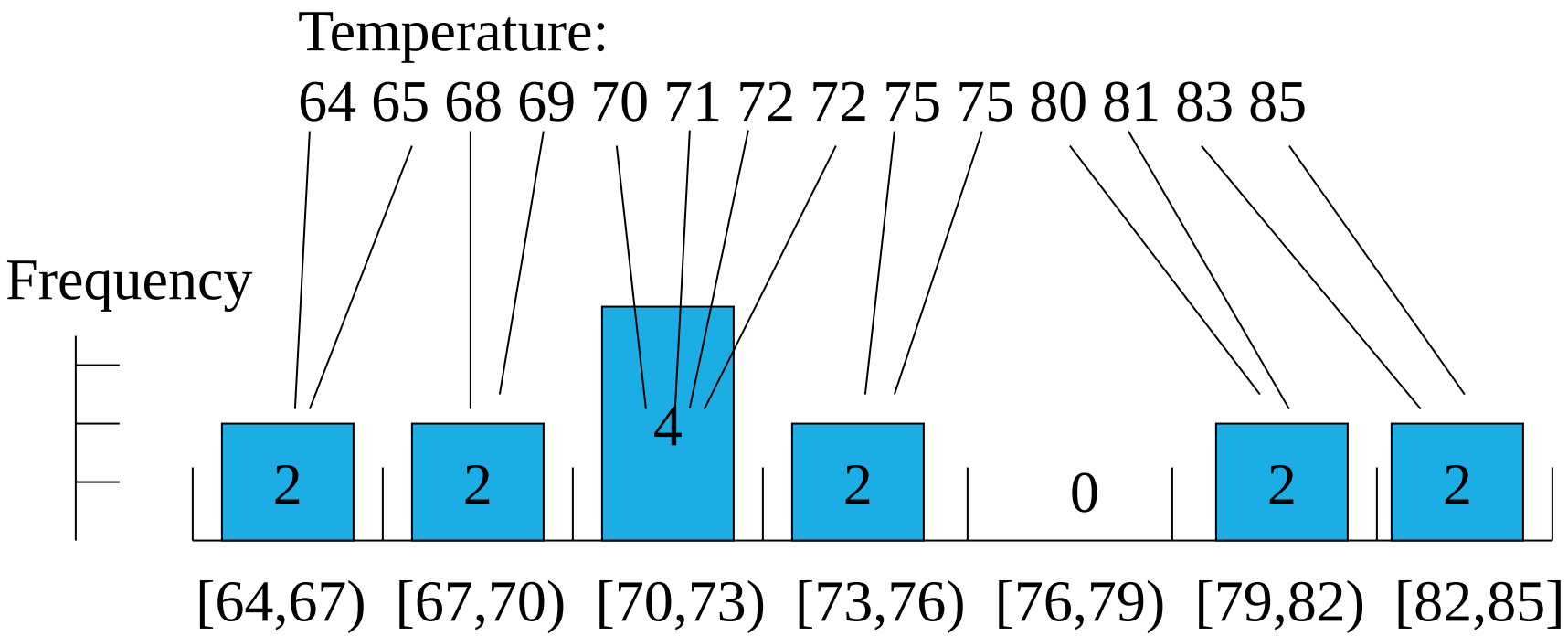
DISCRETIZATION - STAGES



DISCRETIZATION ALGORITHMS TYPES

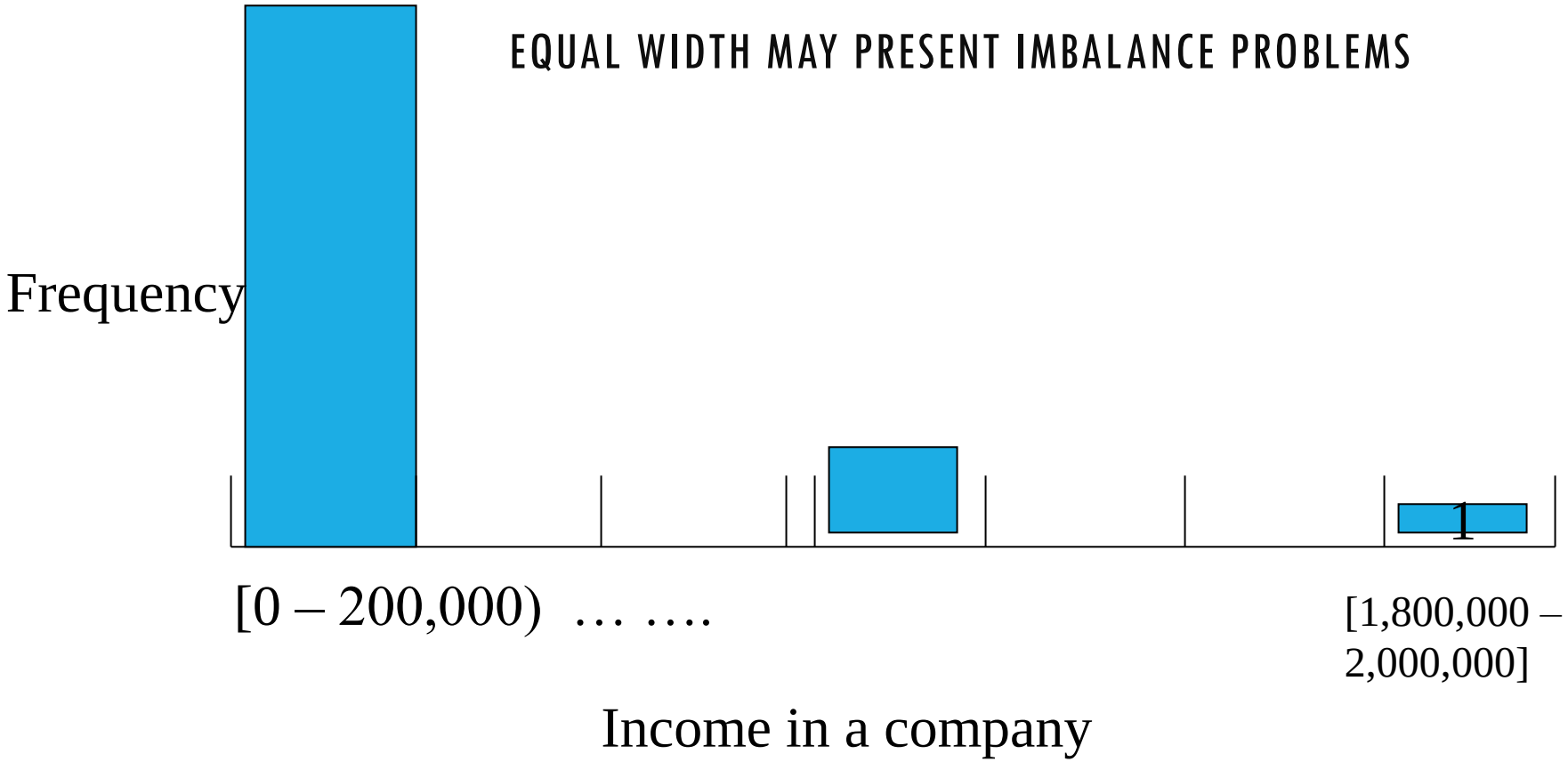
- **Supervised vs. Unsupervised**: Consider the objective (label) attribute or not.
- **Dynamic vs. static**: While building the model or not.
- **Local vs. Global**: Focused on one sub-region of the instance space or considering all of them.
- **Top-down vs. Bottom-up**: Starting with an empty or full list of cut-off points.
- **Direct vs. Incremental**: Use a subsequent optimization process or not.

UNSUPERVISED DISCRETIZATION



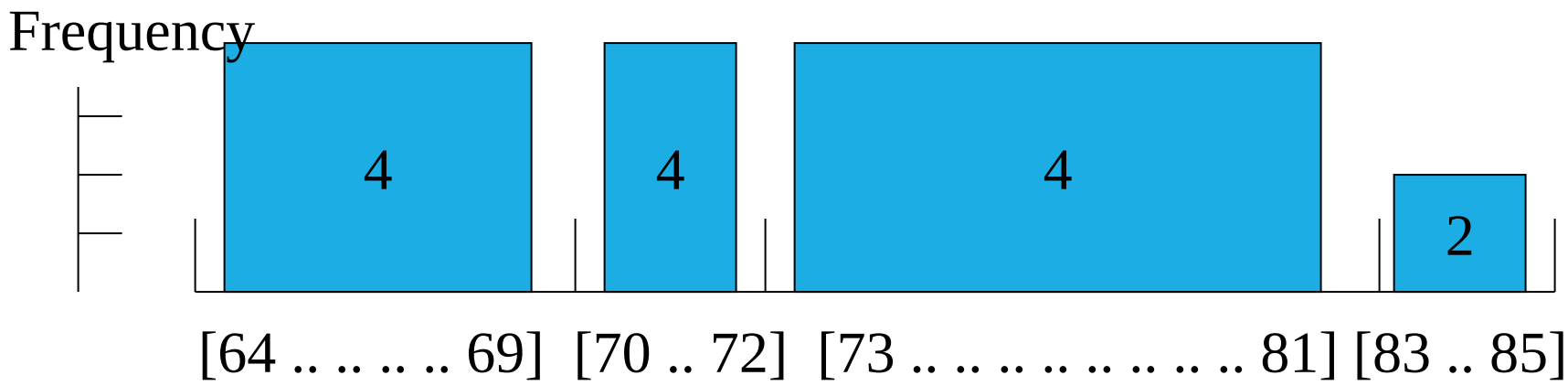
Equal width

UNSUPERVISED DISCRETIZATION



UNSUPERVISED DISCRETIZATION

Temperature values
64 65 68 69 70 71 72 72 75 75 80 81 83 85



Equal frequency (height) = 4, but last box/interval

SUPERVISED DISCRETIZATION - FAYYAD

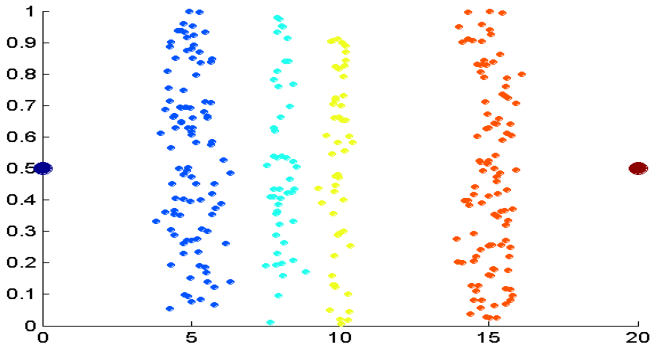
- They begin with the cut-off points given between examples of different classes:

11	14	15	18	19	20	21	22	23	25	30	31	33	35	36
R	C	C	R	C	R	C	R	C	C	R	C	R	C	R

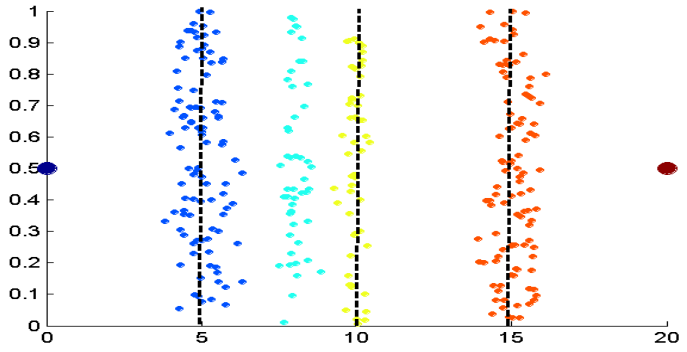
- Minimum Description Length Principle (MDLP), based on entropy, is used to choose useful cutting points from the above and to stop the search.
- MDLP is formulated as the problem of finding the cost of communication between an emitter and a receiver. It is assumed that the sender has the set of instances while the receiver has the class labels.
- A breakpoint induced partition is said to be accepted if and only if the cost of the message required to be sent before partitioning is greater than that required after partitioning.

U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. Proc. 13th Int. Joint Conf. AI (IJCAI-93), 1022-1027. Chamberry, France, Aug./ Sep. 1993.

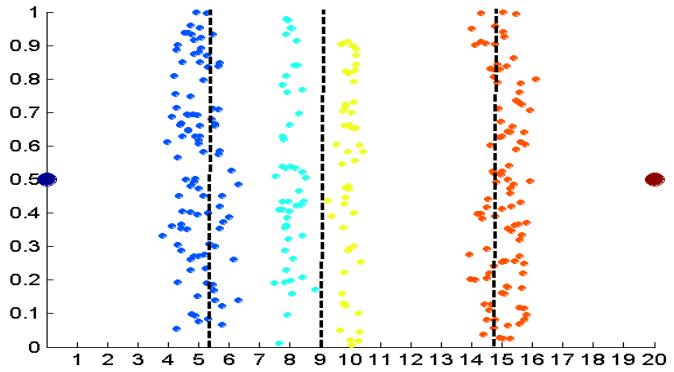
UNSUPERVISED DISCRETIZATION



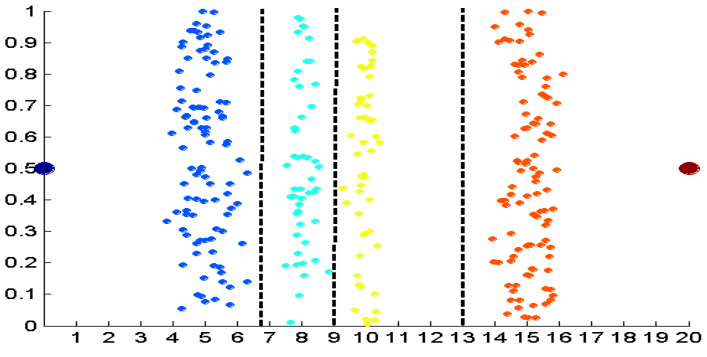
original



Equal width

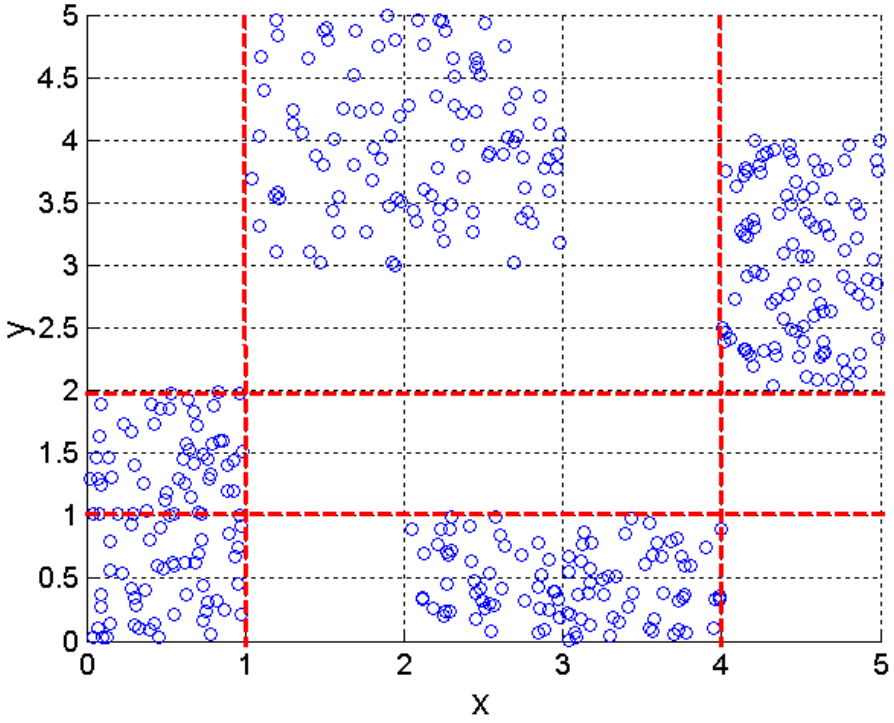


Equal frequency

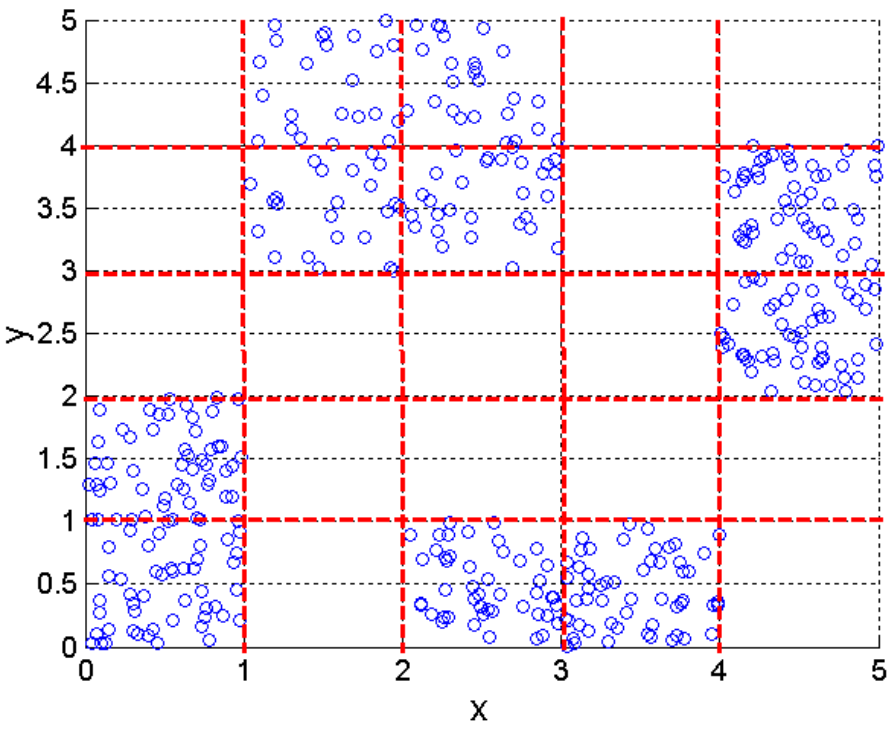


K-means

SUPERVISED DISCRETIZATION - FAYYAD



3 intervals per feature/axis



5 intervals per feature/axis

CAIM DISCRETIZATION

CAIM discretization criterion

$$CAIM(C, D \mid F) = \frac{\sum_{r=1}^{n} \frac{\max_r^2}{M_{+r}}}{n}$$

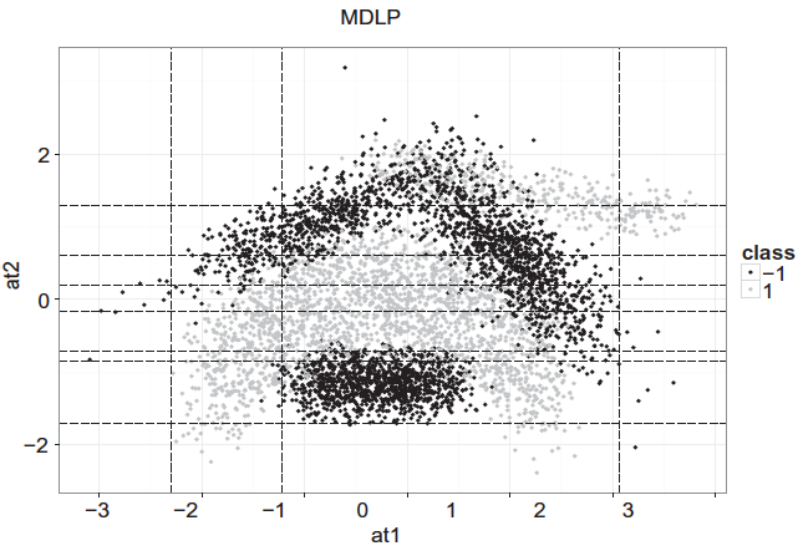
The larger the value of the CAIM ([0, M], where M is # of values of attribute F, the higher the interdependence between the class labels and the intervals

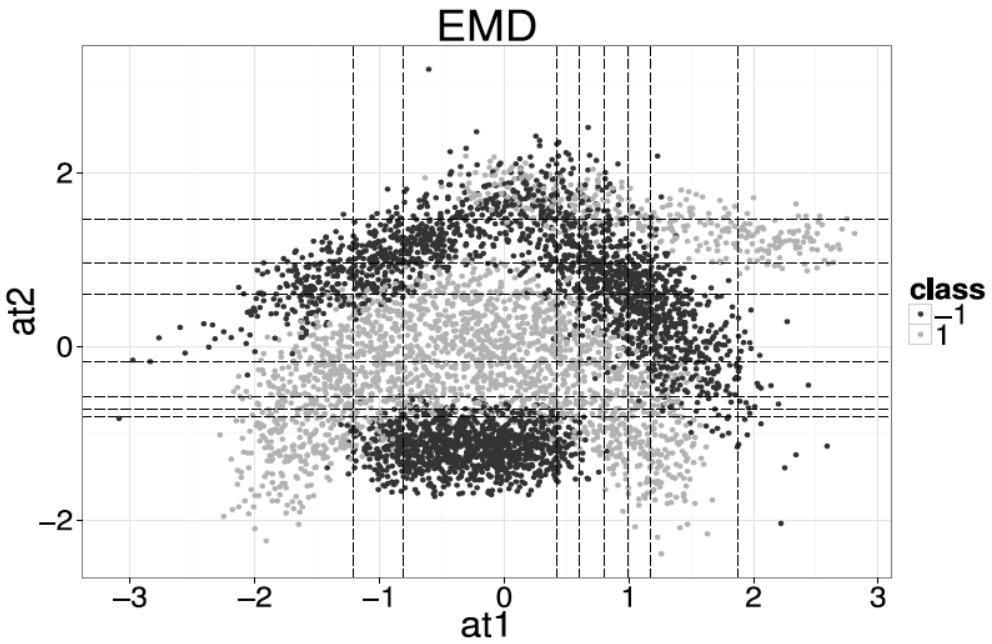
The algorithm favors discretization schemes where each interval contains majority of its values grouped within a single class label (the *maxr* values)

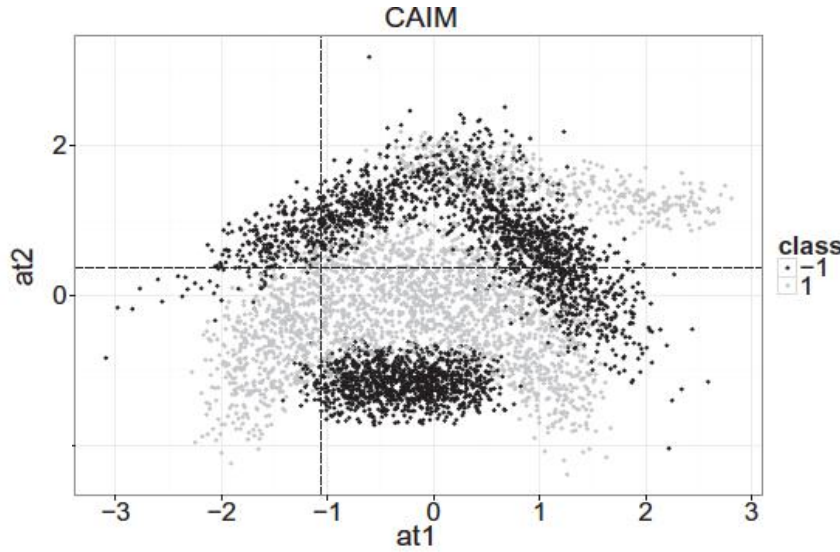
The squared *maxr* value is scaled by the *M+r* to eliminate negative influence of the values belonging to other classes, on the class with the maximum number of values, on the entire discretization scheme

The sum is divided by the number of intervals (*n*) to favor discretization schemes with small number of intervals

SUPERVISED DISCRETIZATION





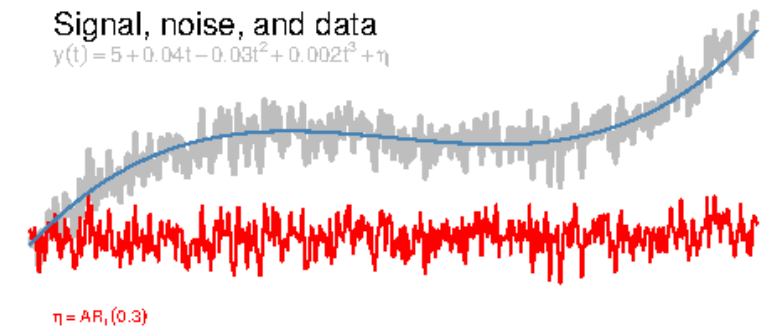


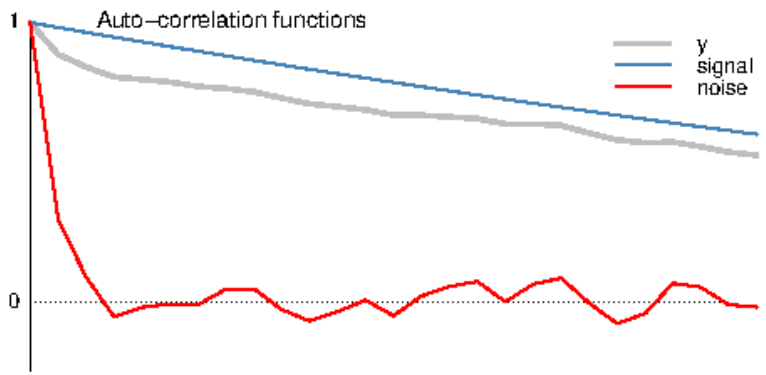
FINAL THOUGHTS

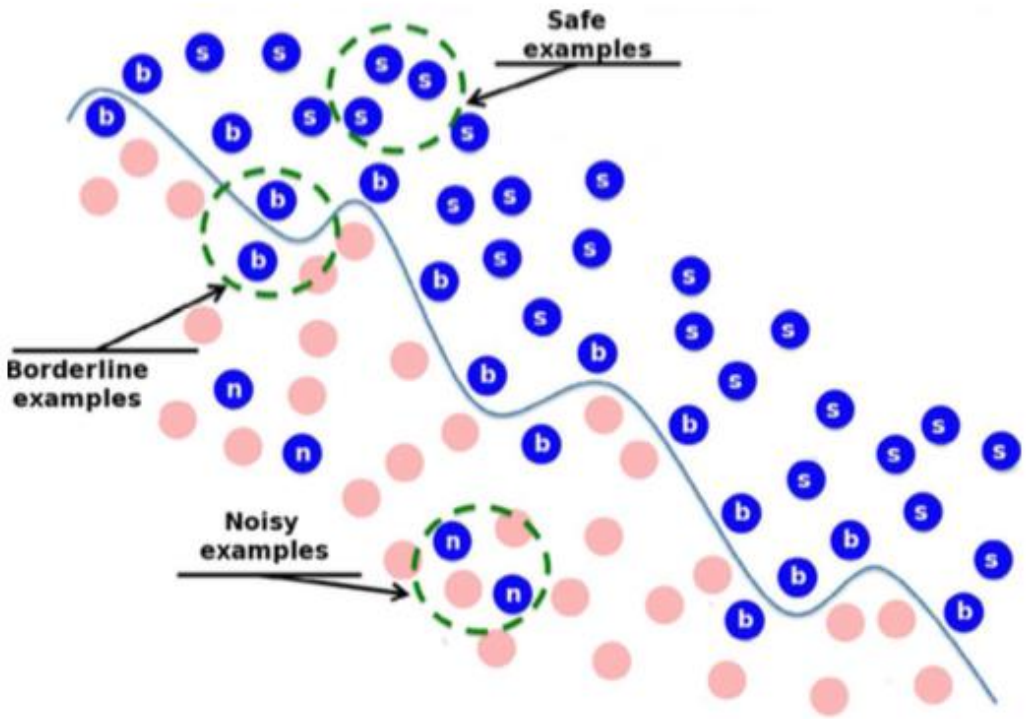
- Which discretizer will be better?
  - As always, it will depend on the application, user needs, etc.
- What is a good discretization?
  - Total number of intervals
  - Number of inconsistencies caused
  - Predictive hit rate

NOISE DATA | 6

NOISE DATA IS MORE THAN PURE SIGNAL NOISE

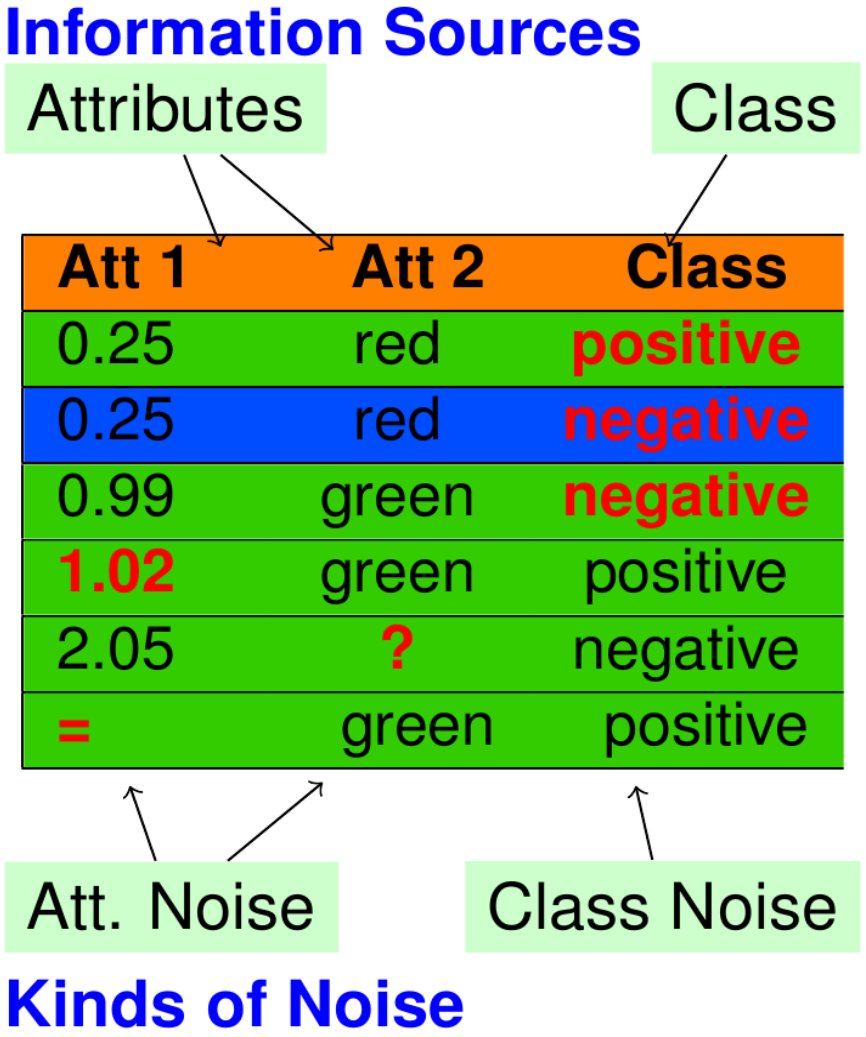




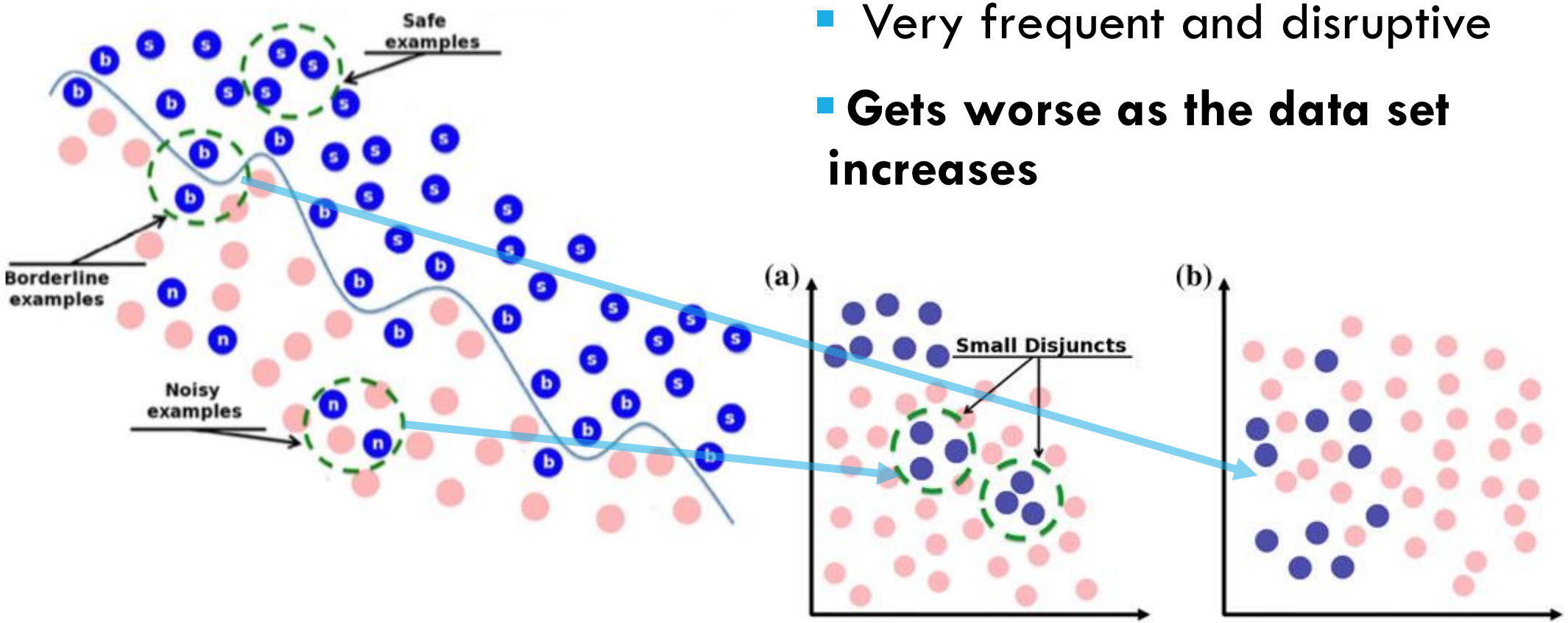


TYPES OF NOISE

- Attribute noise vs. class noise
- Distribution of the noise?
- Is the noise dependent on the original value?

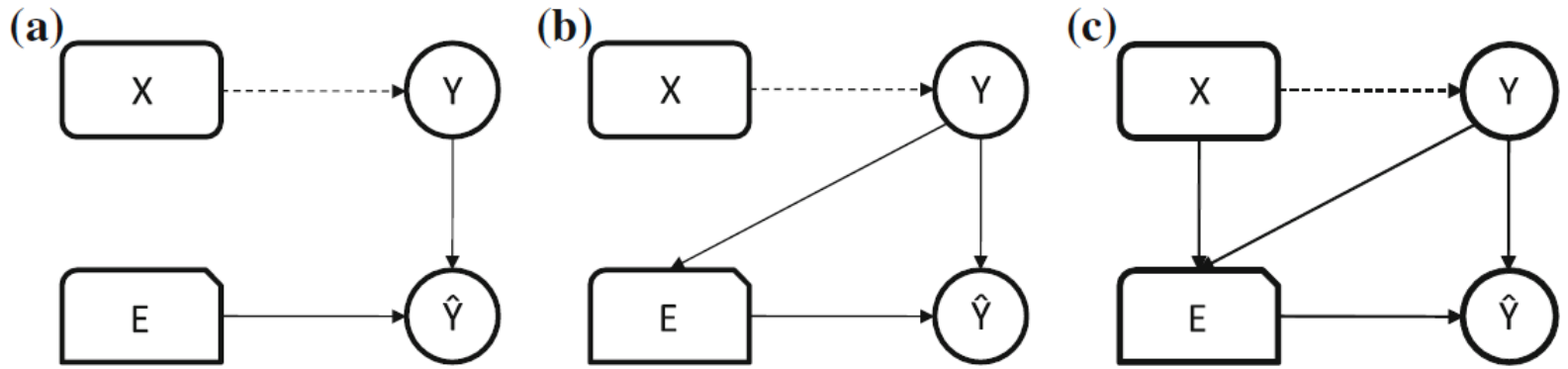


NOISE DATA



- Very frequent and disruptive
- **Gets worse as the data set increases**

NOISE INTRODUCTION MECHANISMS



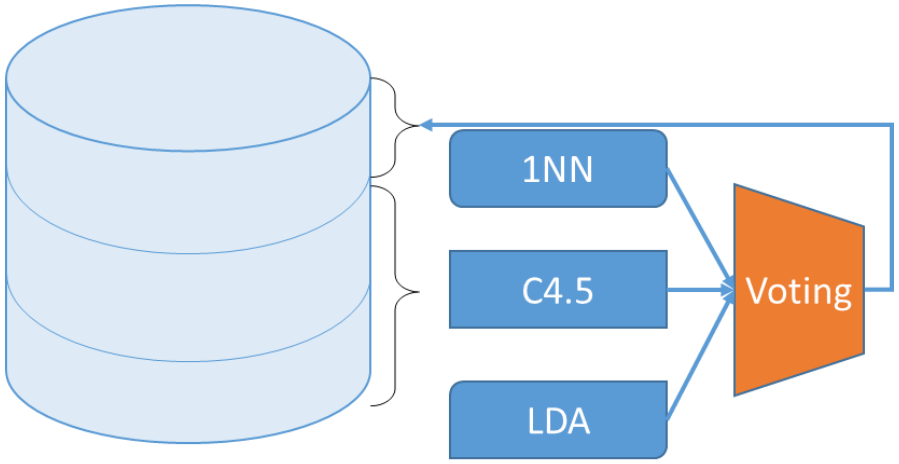
a) NCAR → noise completely at random
b) NAR → noise at random
c) NNAR → noise not at random

Can be treated in preprocessing

TYPES OF NOISE

- **Attribute noise** is more harmful than class noise
  - Attribute noise is more harmful in those attributes that are highly correlated with the class
- Removing or correcting examples → could improve performance.
- Most of the work in the literature focuses only on class noise 😢
  - Re-utilization of instance selection algorithms (ENN, CNN…)
  - Developing new algorithms

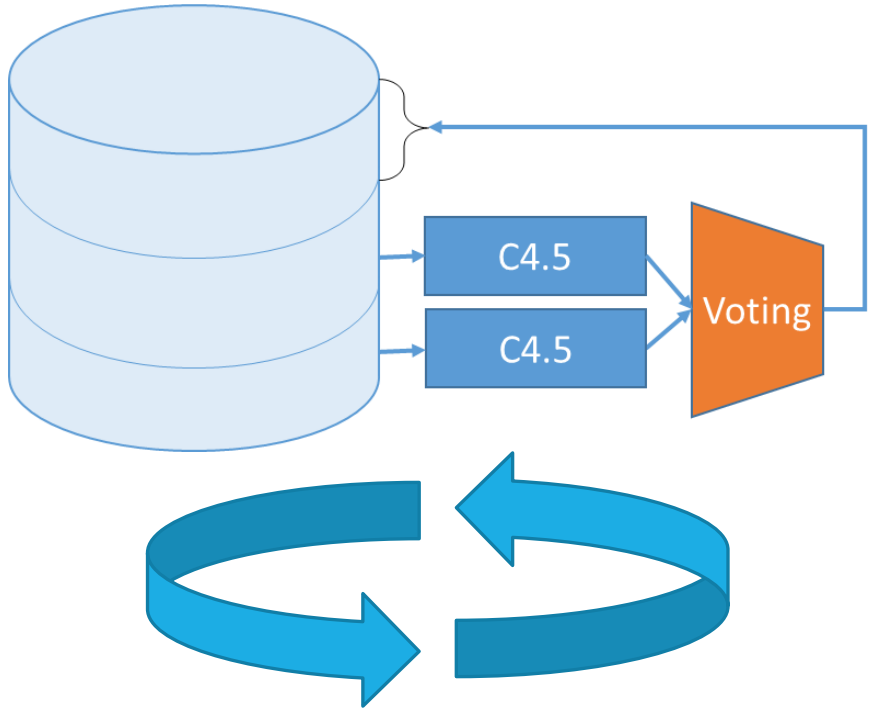
ENSEMBLE FILTER (EF)



- Partitions the training data in k-folds
- A set of classifiers (e.g. C4.5, 1NN and LDA) are used to build a model in a k-fcv fashion
- A voting is yield for each instance → if failed then the instance is removed

C.E. Brodley, M.A. Friedl, Identifying Mislabeled Training Data, Journal of Artificial Intelligence Research 11 (1999) 131–167.

ITERATIVE PARTITIONING FILTER (IPF)



- EF does not take advantage from the cleaned data set it creates
- If we use the cleaned data set to filter again, more accurate decisions will be made
- IPF is iterative:
  1. IPF partitions the data set in *k* folds and trains *k* classifiers (usually C4.5)
  2. Each instance is voted by the *k-1* classifiers as noisy
  3. If the amount of removed instances is large enough **goto 1**

T.M. Khoshgoftaar, P. Rebours, Improving software quality prediction by noise filtering techniques, Journal of Computer Science and Technology 22 (2007) 387–396.

COMPARISON AMONG THE NOISE FILTERS

- We induced 10% of noise in the banana data set
  - We will focus on one region for the sake of clarity
- After applying the filters, it can be seen that many grey dots are removed from the core hollow dot section
- Black dots are also removed from the upper right part
- IPF tends to remove more examples due to its iterative nature

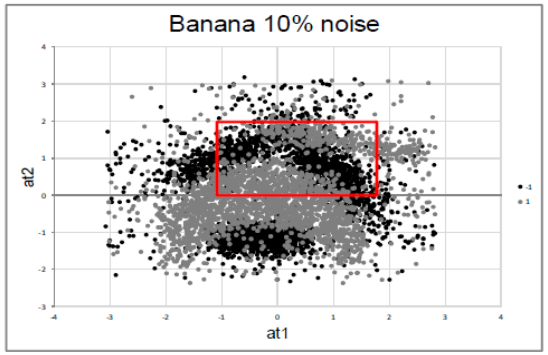


Figure 8: Banana with 10% noise. The red rectangle indicates the region used in the subsequent figures

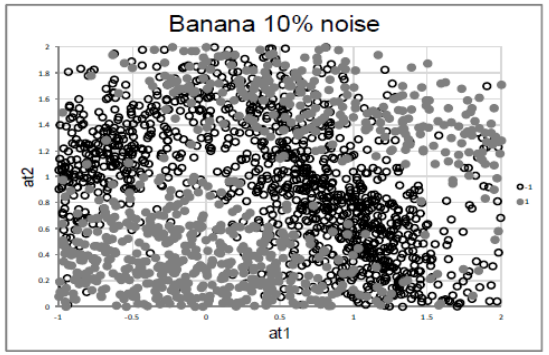


Figure 9: Banana with 10% noise zoomed in an region with class overlapping

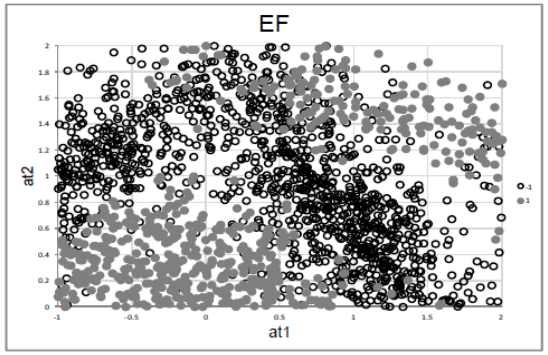


Figure 10: Noisy banana filtered with EF in the zoomed region

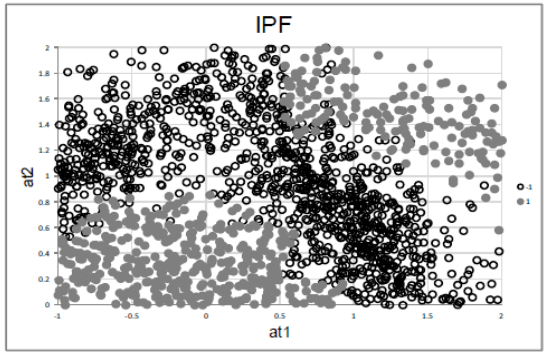
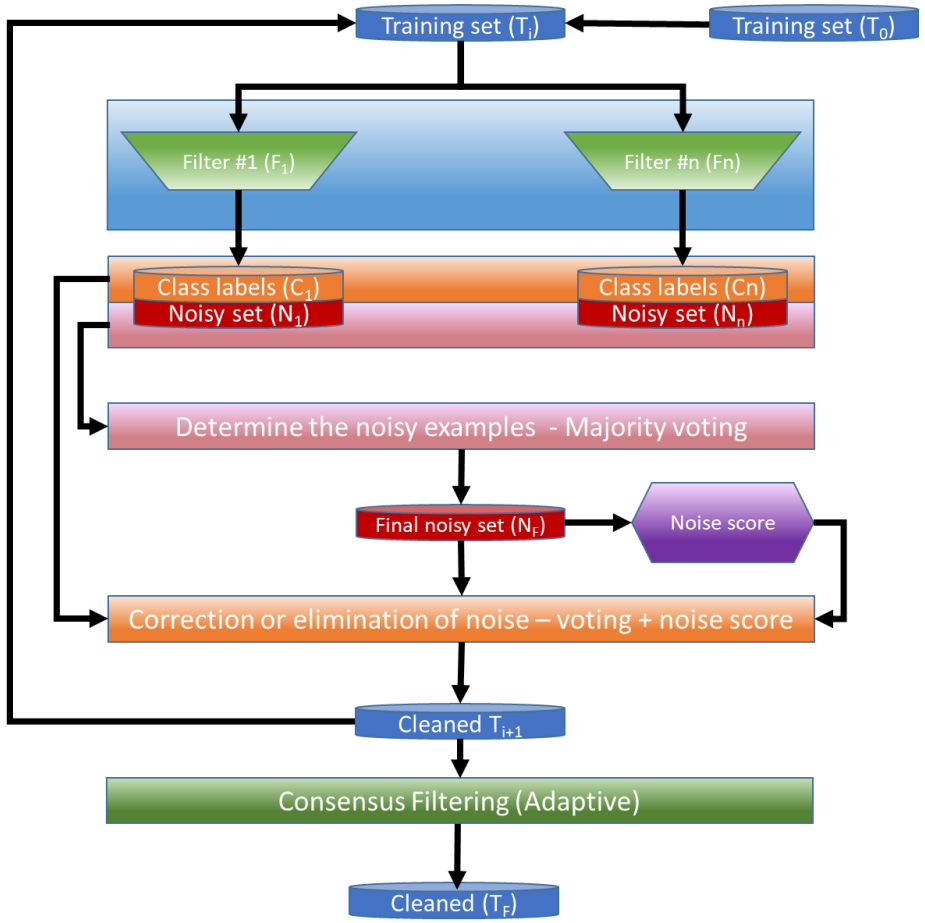


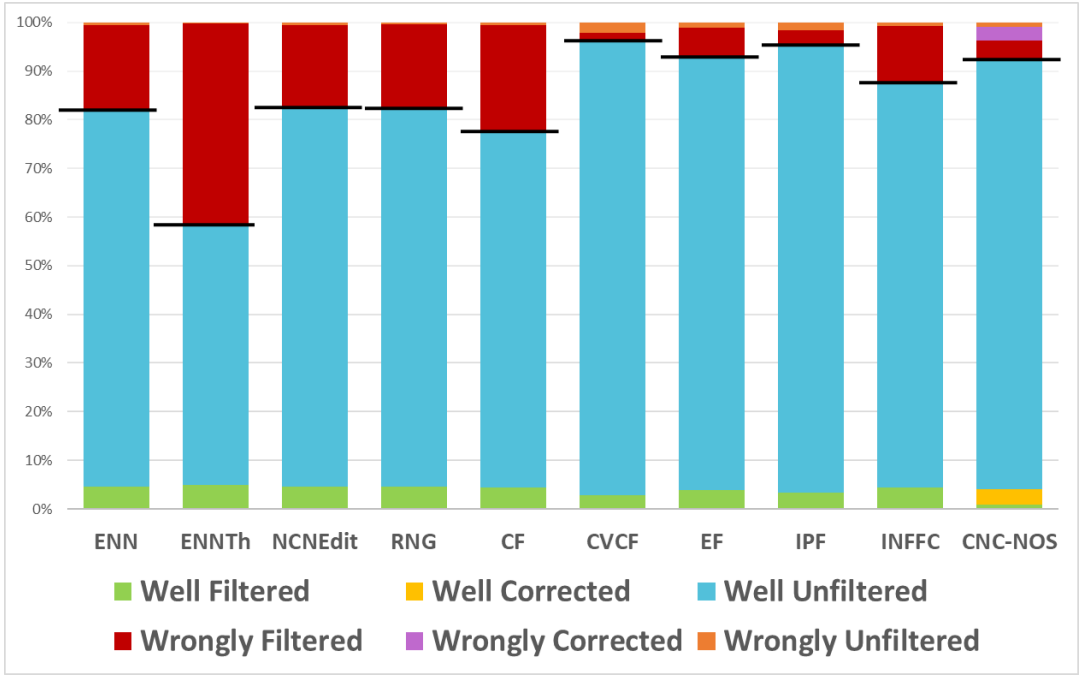
Figure 11: Noisy banana filtered with IPF in the zoomed region

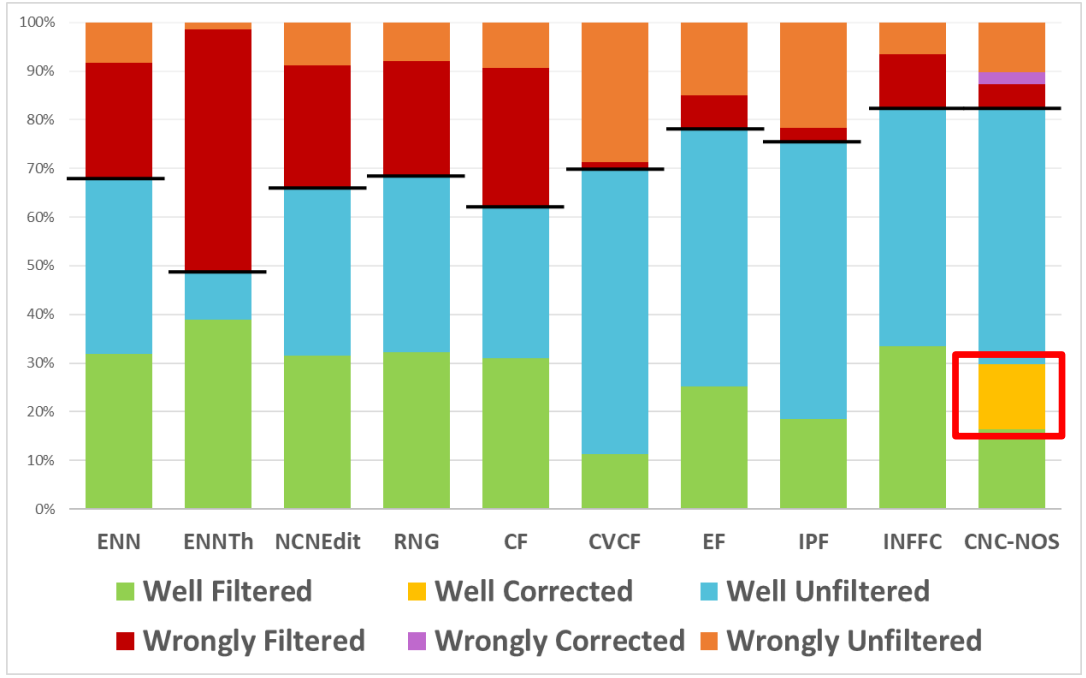
DATA REPARATION: CNC-NOS



- Idea: take several noise filters over the same data
- Examine what they predict on noise instances
  - If they agree on a different output → repare output
  - If the do not agree → filter instance

DATA REPARATION: CNC-NOS



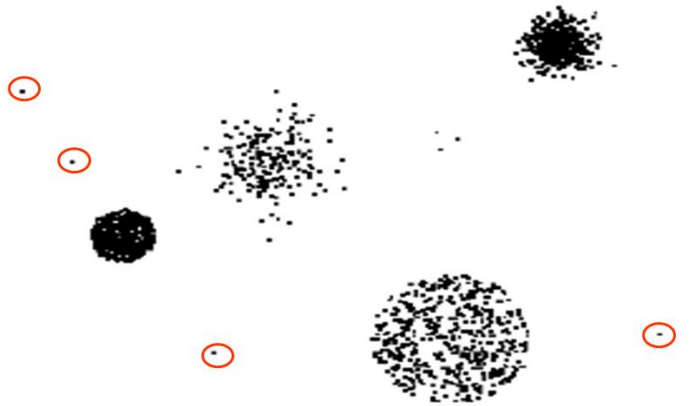


5% noise

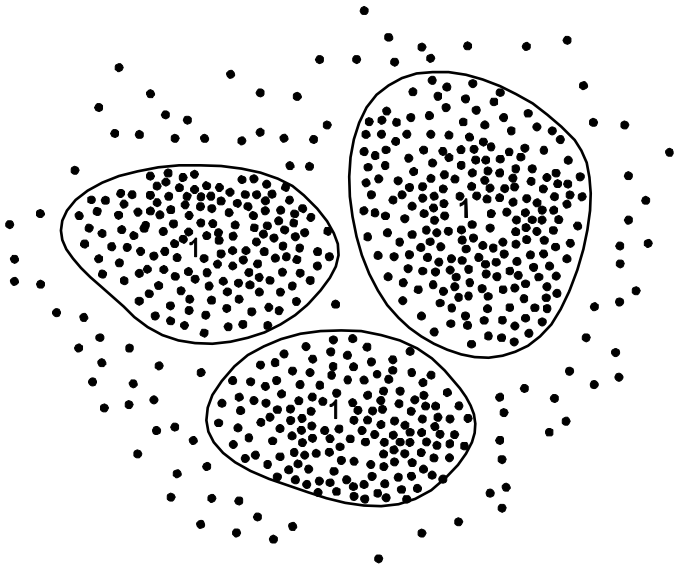
40% noise

NOISE DATA IS NOT OUTLIER DATA

- Abnormal, outliers or extreme values: they are correct even if they are statistically abnormal.
- They can be a drawback for methods based on weight adjustment (e.g. AANN).



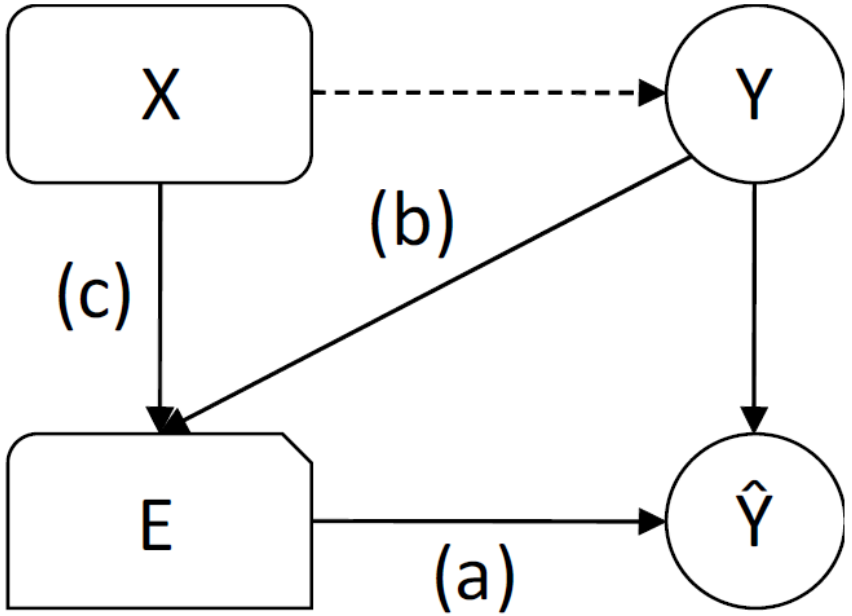
- Detection techniques:
  - Define a distance and see the individuals with greater distance average to the rest of the individuals.
  - Partial clustering: the data are grouped into clusters and the data out can be considered outliers.



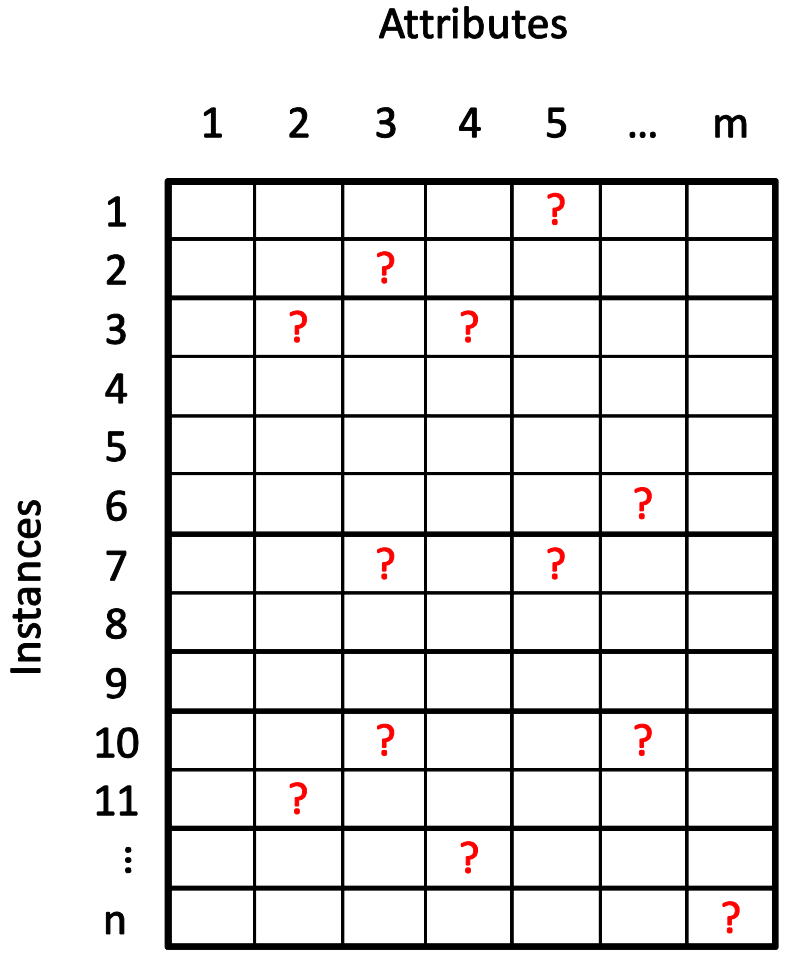
MISSING VALUES | 7

MISSING VALUES

- What causes the missingness?



a) MCAR
b)MAR
c) MNAR



MISSING AT RANDOM (MAR)

- The distribution of B (presence of missing value $\hat{Y}$) should be related to X and some unknown parameters ζ, so we have a probability model for B described by $P(B \mid X, \zeta)$.
- Having missing at random (**MAR**) assumptions means that this distribution is not dependent on X_{mis}:

$$P(B|X_{obs}, X_{mis}, \zeta) = P(B|X_{obs}, \zeta).$$

MISSING COMPLETELY AT RANDOM (MCAR)

- MAR suggests that the not lost values constitute another possible sample from the probability distribution.
  - This condition is known as missing completely at random (MCAR).
- MCAR is a special case of MAR in which the distribution of an example that has a missing value for an attribute does not depend on the data, whether or not it is observed

$$P(B|X_{obs}, X_{mis}, \zeta) = P(B|\zeta)$$

MISSING NOT AT RANDOM (MNAR)

- A third case occurs when MAR is not applicable because the lost values depend on both: the rest of the observed values and the own value.

$$P(B|X_{obs}, X_{mis}, \zeta)$$

- This model is called not missing at random (**NMAR**) or missing not at random (**MNAR**) in the literature.
  - The only way to obtain a reliable estimator is to model the way in which the data are lost.
  - This is a very complex task in which a model must be closed on account of the missing values that should then be incorporated into an even more complex model that will be used to estimate the missing values.

DEALING WITH MISSING VALUES

- Discard examples with missing values
  - May induce bias if not MCAR
- Convert lost values to a new value
  - Greater difficulty in interpreting the knowledge extracted
- Imputation methods
  - Posterior methods are unmodified
  - Parametric (MNAR) vs. No parametric (MAR and MCAR)

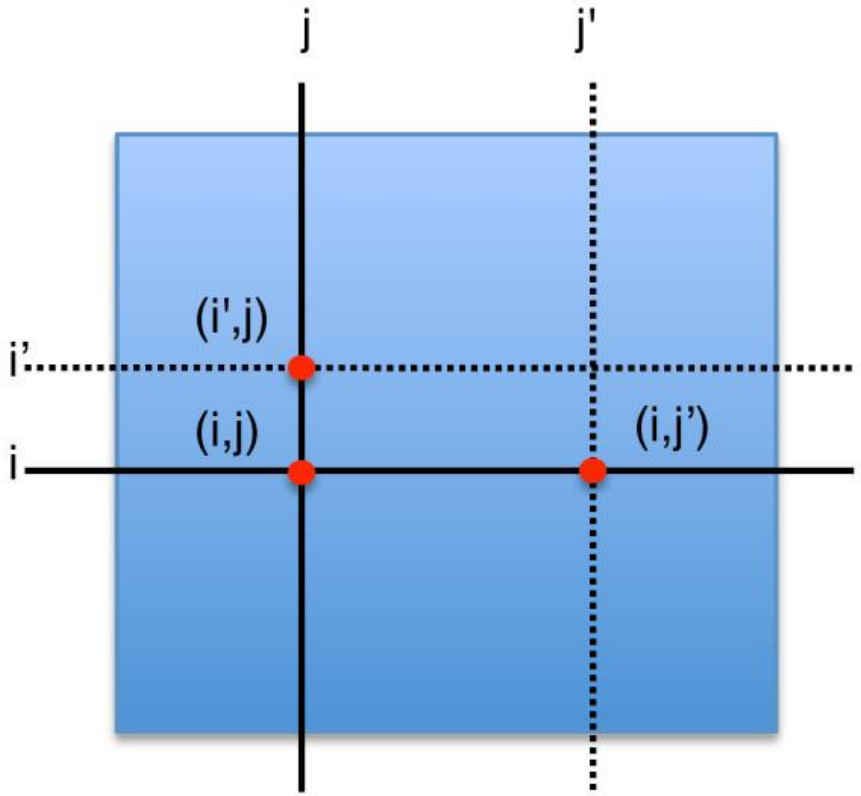
SIMPLE IMPUTATION EXAMPLE

Index	Original value	Missing value	Preserve mean	Preserve std
1	0.0886	0.0886	0.0886	0.0886
2	0.0684	0.0684	0.0684	0.0684
3	0.3515	0.3515	0.3515	0.3515
4	0.9875	0.9875	0.9875	0.9875
5	0.4713	0.4713	0.4713	0.4713
6	0.6115	0.6115	0.6115	0.6115
7	0.2573	0.2573	0.2573	0.2573
8	0.2914	0.2914	0.2914	0.2914
9	0.1662	0.1662	0.1662	0.1662
10	0.4400	0.4400	0.4400	0.4400
11	0.6939	????	**0.3731**	**0.6622**
Media	0.4023	0.3731	0.3731	0.3994
SD	0.2785	0.2753	0.2612	0.2753
Estimation error			0.3208	**0.0317**

NON PARAMETRIC IMPUTATIONS – MAR/MCAR

- Estimation by means of automatic learning techniques:
  - Imputation with k-NN (**KNNI**)
  - Imputation with k-NN and weights (**WKNNI**)
  - Cluster-based Imputation (**KMI**)
  - Imputation based on SVM algorithms (**SVMI**) or any other ML algorithm
  - Others: event covering (**EC**), singular value decomposition (**SVDI**), local least squares imputation (**LLSI**)
- How? Use the other input attributes to generate a model that predicts on the missing attribute

KNN IMPUTATION (KNNI)



- Very popular and simple approach:
  - Only needs a similarity function between examples
  - Can deal with mixed-type attributes
- Pick the mean or mode from the neighbor(s) for the missing values (i' and j')

G. Batista, M. Monard, An analysis of four missing data treatment methods for supervised learning, Applied Artificial Intelligence 17 (2003) 519–533.

PARAMETRIC IMPUTATIONS - MNAR

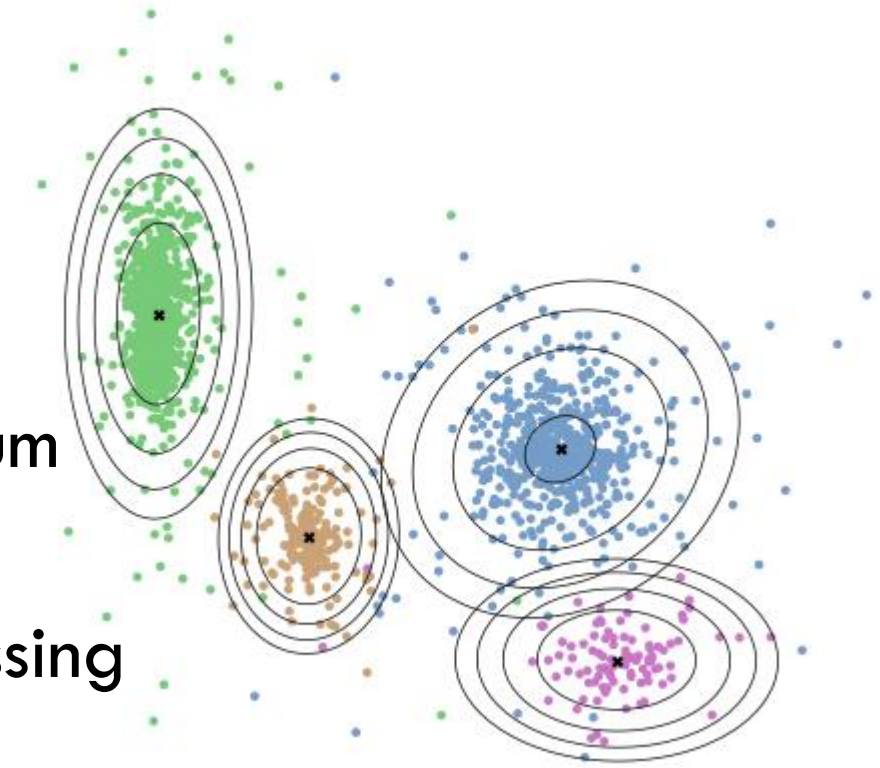
- We may know the underlying data distribution for the simples
  - For instance, the attributes are generated by a Gaussian distribution

- Instead of estimating the value to impute, we:

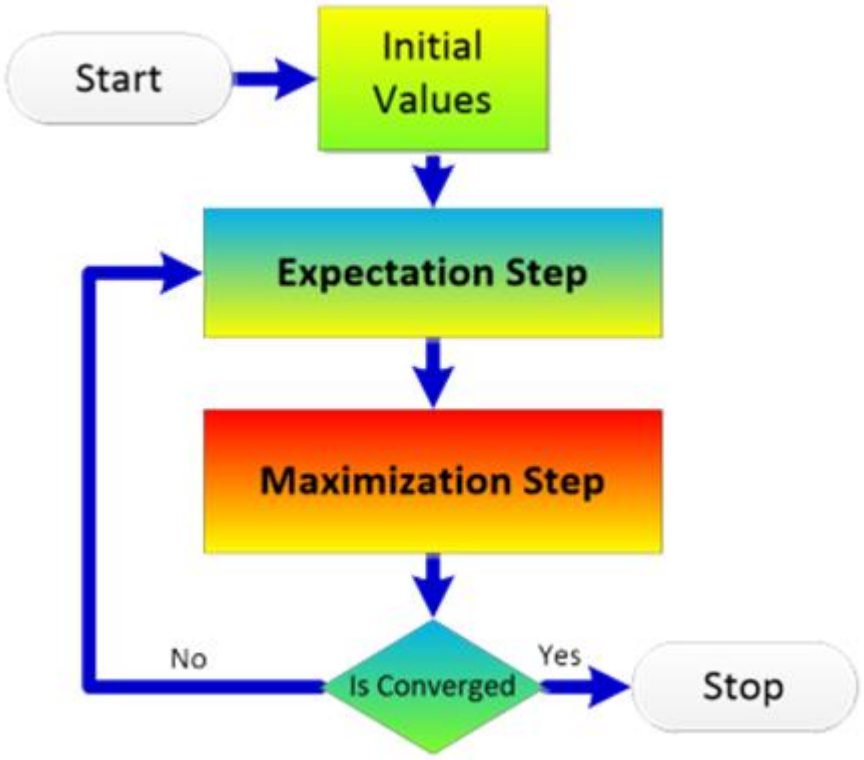
1. Identify the parameters of the underlying distribution
2. Estimate the parameters with max likelihood with our examples
   - (Extra) If we know the distribution of missing data with respect to the actual values, then that information should be included in the model as well
3. Draw values from the distribution to impute the data

EXPECTATION-MAXIMIZATION

- Iterative process to obtain the distribution parameters
- It converges to the parameters with maximum likelihood to our simples
- We need to provide initial values to the missing values (average ob values are usual)
- **Drawback**: the maximization phase mostly defined to well known statistical distributions → GMM



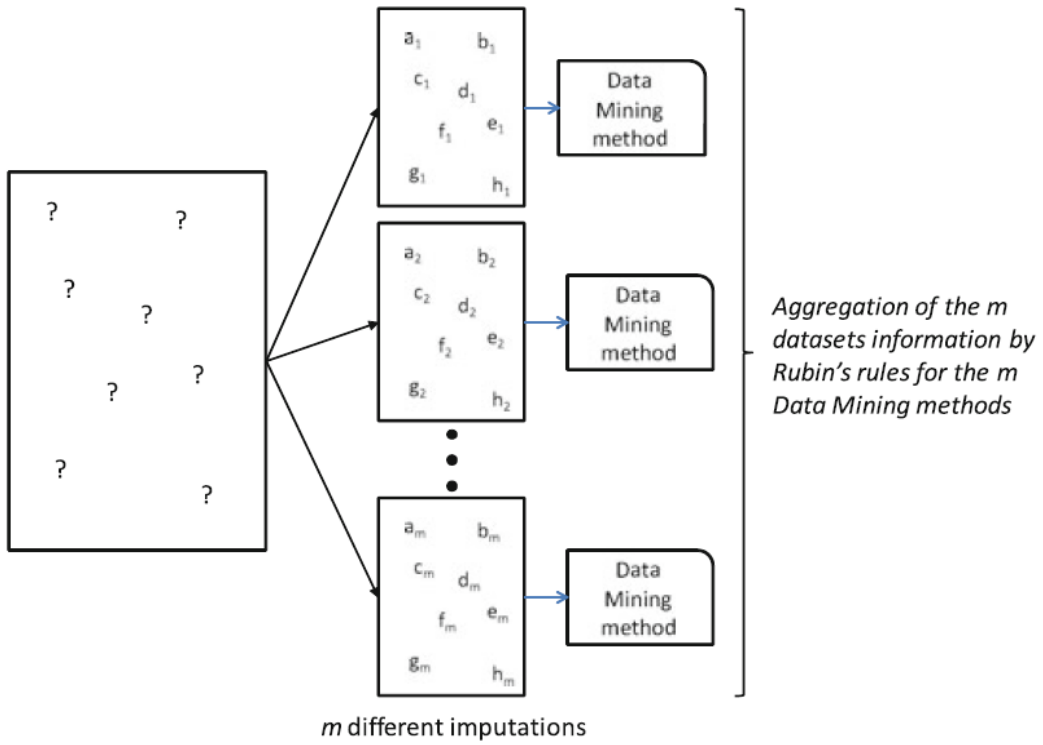
EXPECTATION-MAXIMIZATION



- We assume a known underlying distribution with **unknown parameters**
  - For example a Gaussian with unknown mean and std. dev.
- EM iterates in 2 steps:
  1. E-step → refines the unknown parameters by drawing samples from the current estimates
  2. M-step →obtains the maximum of the likelihood function for the unknown parameters
- EM will converge to a parameter set for the distribution…
- …and then we impute by using the distribution to sample values or create a statistical model (e.g. ridge regression)

A. Dempster, N. Laird, D. Rubin, Maximum likelihood estimation from incomplete data via the EM algorithm (with discussion), Journal of the Royal Statistical Society, Series B 39 (1977) 1–38.

MULTIPLE IMPUTATION (MI)



- EM underestimate standard errors
- MI is a Monte Carlo approach → it models a Bayesian posterior distribution instead the data distribution
- Several samples are thus drawn for a single missing value → several version of the same data set
  - Each data version is treated separately…
  - …and the results are later aggregated

D.B. Rubin, Multiple Imputation for Nonresponse in Surveys, Wiley, 1987.

COMPARISON

- By taking a portion of the banana data set, we removed some values
- The three most typical imputation algorithms were used to recreate the instances with missing values
  - We used the complete data set to do so
  - In the case of EM, we used a ridge regression procedure (commonly employed in statistics)
- The presence of MVs deals great impact in the information present in the data
  - But since banana has only 2 features this effect is exacerbated

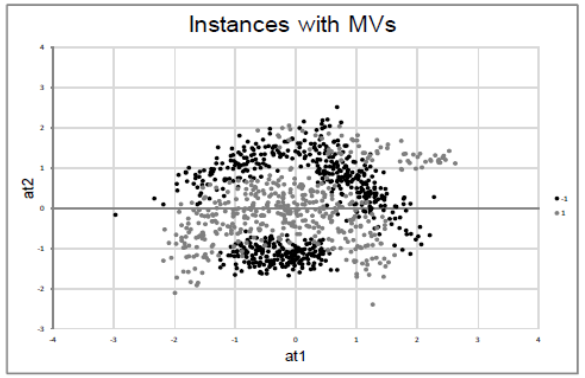


Figure 4: Banana's instances affected by MVs (still depicted as complete)

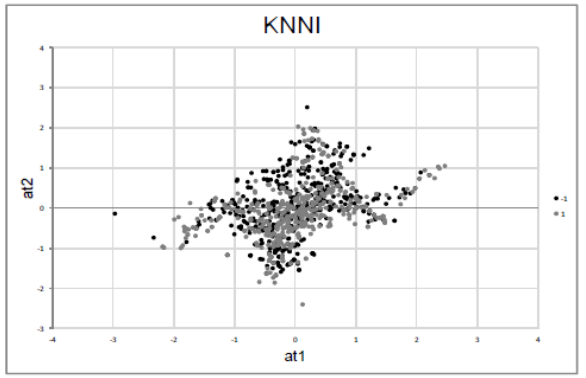


Figure 5: Banana imputed values with kNNI

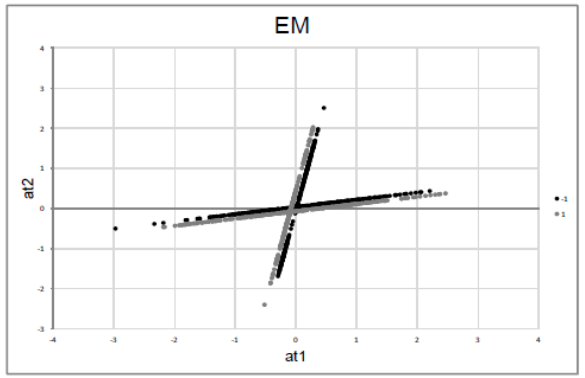


Figure 6: Banana imputed values with EM

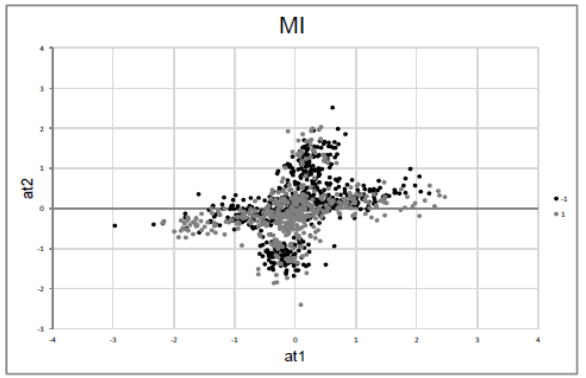
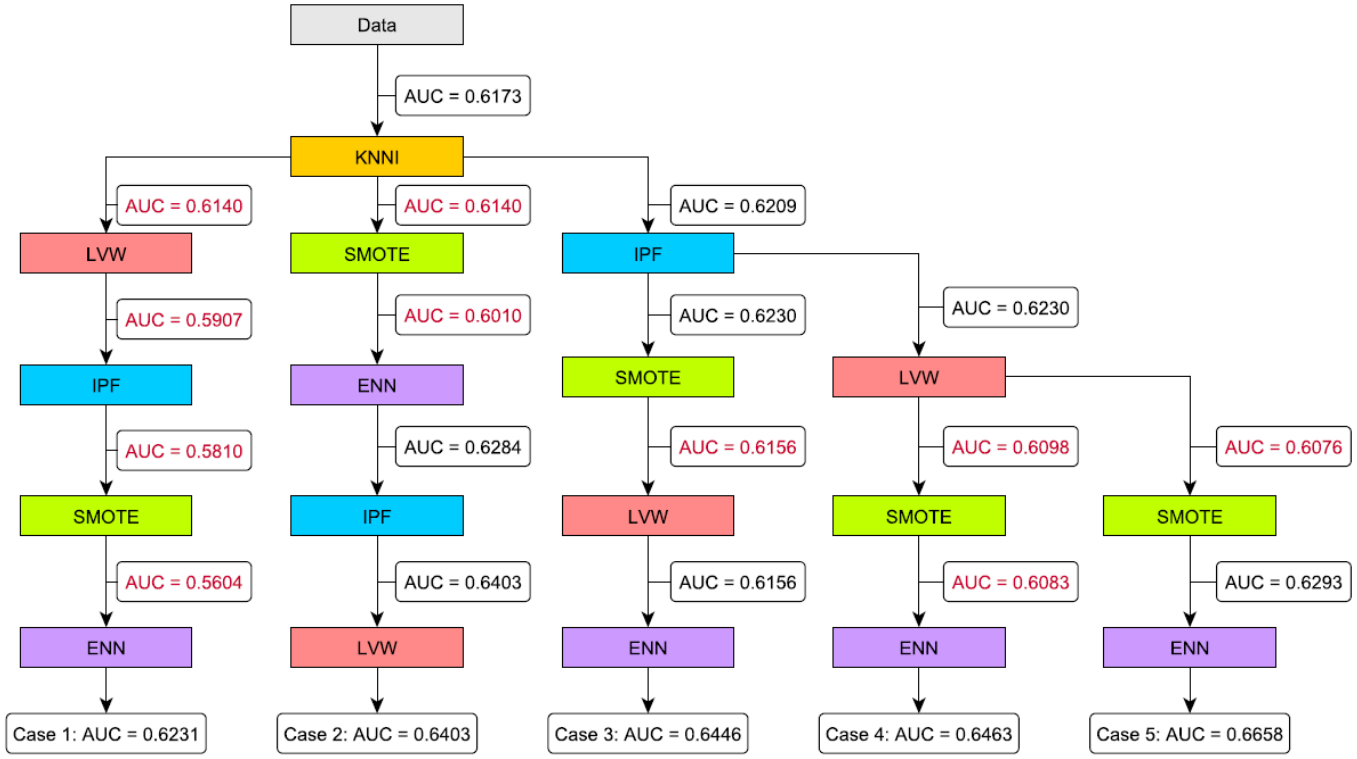


Figure 7: Banana imputed values with MI

CONCLUSIONS | 8

IMPORTANCE OF PREPROCESSING ORDER

- In real problems, several problematics concur in the data
  - We may face MVs, noise, redundant features, excess of examples, etc.
- The question: Which order should I consider to apply my preprocessing algorithms?
- Sometimes the order is forced
  - E.g. most preprocessing algorithms cannot deal with MVs → imputation comes first
- Sometimes we must explore the possibilities or rely in expert knowledge
- We depict the AUC results over the ECDBL'14 competition data set for 5 cases (configurations)
  - Improvements in AUC appear as soon as we choose the correct order to first eliminate noise and the carry on



STILL TO BE EXPLAINED: DATA INTEGRATION

- We did not tackle any aspects of data integration, as such as:
  - Schema integration (as in Databases)
  - Duplicate or inconsistent records
  - Redundancy elimination
  - Normalization and data scalation
  - Outlier detection and removing
  - And many more!

SOFTWARE

01

INSTANCE SELECTION

- http://sci2s.ugr.es/pr/index.php

02

FEATURE SELECTION / TRANSFORMATION

- Fselector
- Caret
- Boruta
- dimRed

scikit learn

- Feature selection
- Manifold learning

03

DISCRETIZATION

- discretization

scikit learn

- Discretization

- pyCAIM
- Mdlp-discretization

04

NOISE DATA

- noiseFiltersR

05

MISSING VALUES

- https://cran.r-project.org/web/views/MissingData.html

scikit learn

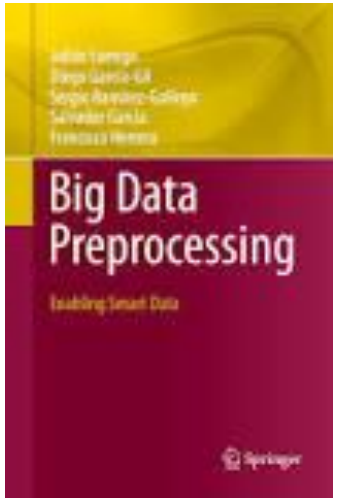
- Imputation of missing values

SmartData:
a complete approach in
https://cran.r-project.org/web/packages/smartdata/index.html

Oldie but goldie
http://www.keel.es/
480+ preprocessing methods

BIG DATA PREPROCESSING?

- Diego can tell us



Luengo, J., García-Gil, D., Ramírez-Gallego, S., García, S., & Herrera, F. (2020). ***Big Data Preprocessing: Enabling Smart Data***. Springer Nature.