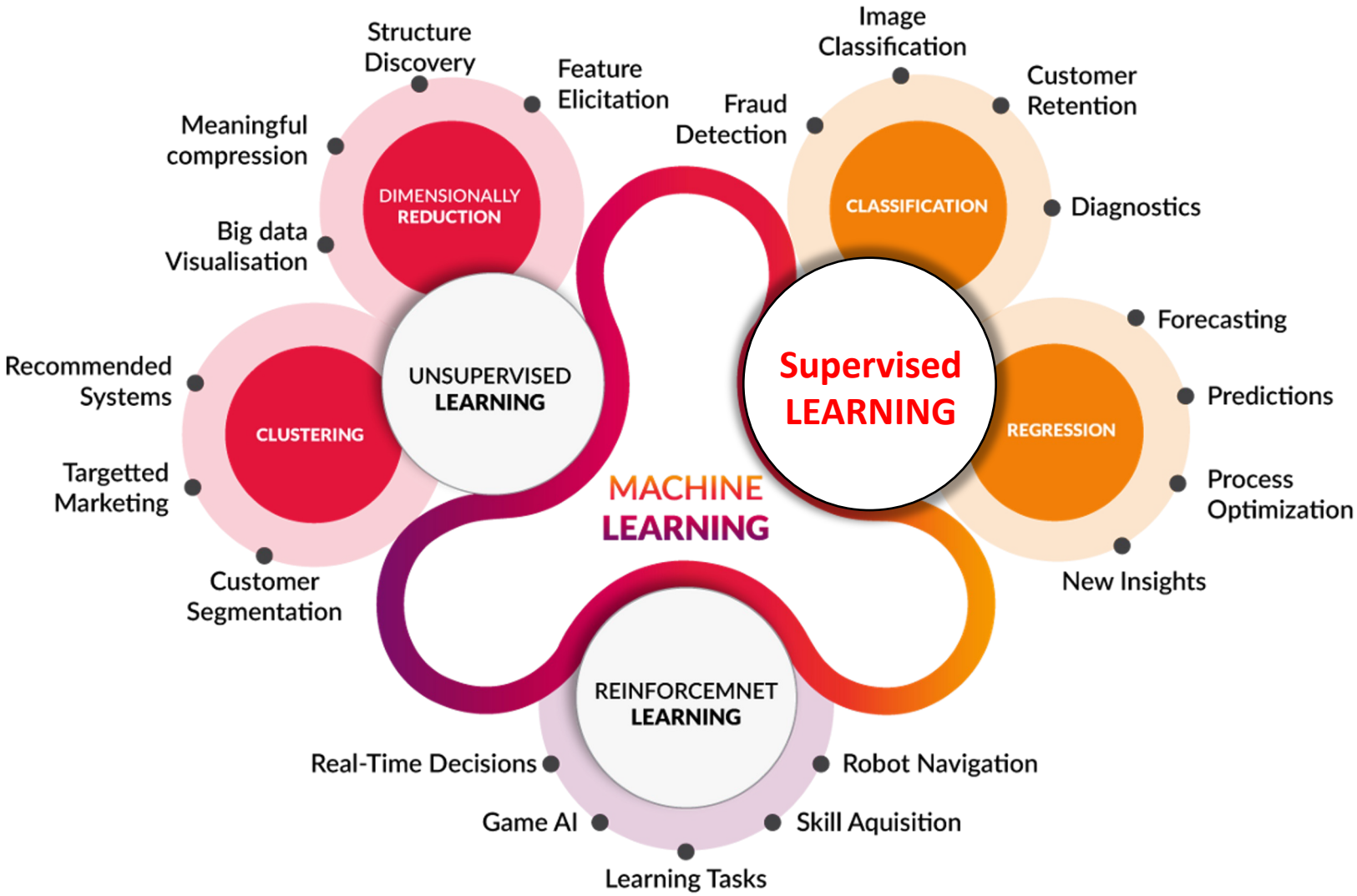

MACHINE LEARNING
UNSUPERVISED LEARNING
DIMENSIONALLY REDUCTION
Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
CLUSTERING
Recommended Systems
Targetted Marketing
Customer Segmentation
Supervised LEARNING
CLASSIFICATION
Image Classification
Customer Retention
Fraud Detection
Diagnostics
REGRESSION
Forecasting
Predictions
Process Optimization
New Insights
REINFORCEMNET LEARNING
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

Interpretability

If a machine learning model performs well, why do not we just trust the model and ignore why it made a certain decision?

→ The problem is that a single metric is an incomplete description of most real-world tasks.

There is no mathematical definition of interpretability. Some options from [Ref. 1]:

- Interpretability is the degree to which a human can understand the cause of a decision.
- Interpretability is the degree to which a human can consistently predict the model's result.

There are Machine Learning model that can be mostly interpreted (e.g. Linear regression and Decision Tree) and there are approaches to study models as a black-box (e.g. Shapley values [Ref. 2]).

→ We can use Mathematical Optimization to boost the intepretability of Machine Learning (see project 10).

Interpretability: SVM

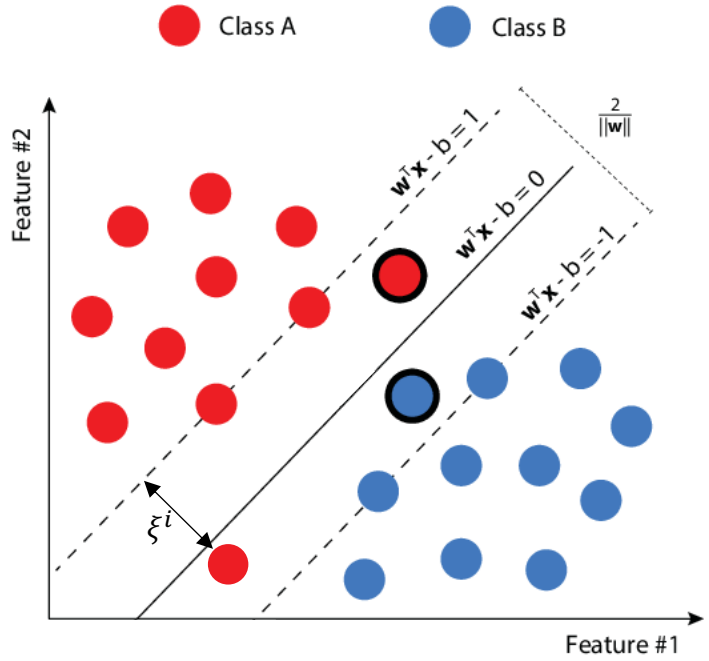
The idea of soft-margin SVM is: *allow SVM to make a certain number of mistakes and keep margin as wide as possible so that other points can still be classified correctly*.

$$\min \boxed{\|\omega\|} + C \sum_{i \in I} \xi^i$$

$$s.t. \quad y_i(\omega^T x_i + \beta) \geq 1 - \xi^i \qquad \forall i \in I$$

$$\xi^i \geq 0. \qquad \forall i \in I$$

$$\omega \in \mathbb{R}^d, \beta \in \mathbb{R}$$



→Can we make this more interpretable?

Yes. If we formulate with a mathematical model, we can binarize or all the features while searching for ω and β.

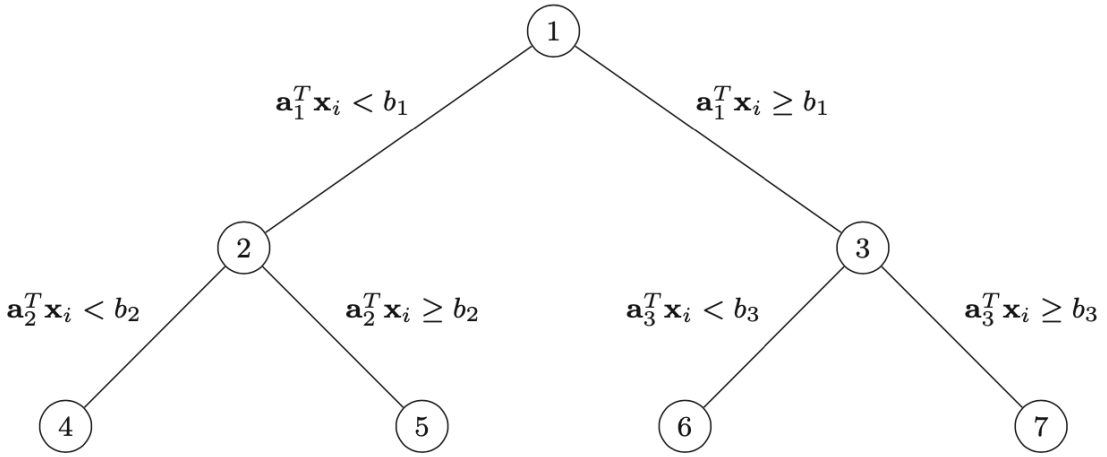
Optimal Classification Tree

The main shortcoming of Decision Trees is its fundamentally greedy nature. Each split in the tree is determined in isolation without considering the possible impact of future splits in the tree.

→If we formulate the tree creation in a MIP, we can solve the problem optimally and achieve better generalization.

Key decisions:

- We must choose when to branch or stop.
- After deciding to stop, we must choose a label to assign to this new leaf.
- After choosing to branch, we must choose which of the variables to branch on.
- We must also choose to which leaf node a point will be assigned such that the structure of the tree is respected.



Optimal Classification Tree

1. **When to split.** We use a binary variable a_{jt} to decide the attribute to split on (univariate decision tree) and a variable b_t to decide the threshold. The set $\mathcal{T}_B$ ($\mathcal{T}_L$) contains all the branching (leaf) nodes. The decision whether to split or not is model with the variable d_t.

$$\sum_{j=1}^{p} a_{jt} = d_t, \quad \forall t \in \mathcal{T}_B, \tag{2}$$

$$0 \leq b_t \leq d_t, \quad \forall t \in \mathcal{T}_B, \tag{3}$$

$$a_{jt} \in \{0, 1\}, \quad j = 1, \ldots, p, \quad \forall t \in \mathcal{T}_B, \tag{4}$$

$$d_t \leq d_{p(t)}, \quad \forall t \in \mathcal{T}_B P \backslash \{1\}, \tag{5}$$

I can be a branching node if my parent is a branching node.

Optimal Classification Tree

2. **Which points to which leaves.** We use a binary variable z_{it} to assign a point to a node. Variable l_t is used to force a minimum number (N_{min}) of points to branch on.

$$z_{it} \leq l_t, \quad t \in \mathcal{T}_B, \tag{6}$$

$$\sum_{i=1}^{n} z_{it} \geq N_{\min} l_t, \quad t \in \mathcal{T}_B. \tag{7}$$

$$\sum_{t \in \mathcal{T}_L} z_{it} = 1, \quad i = 1, \ldots, n. \tag{8}$$

3. **Binary structure.** $A_L(t)$ is the set of all ancestor nodes whose Left branch has been followed on the path from the root to t (A_L is for right branch). Note the presence of two big-Ms.

$$\boldsymbol{a}_m^\top \boldsymbol{x}_i < b_t + M_1(1 - z_{it}), \quad i = 1, \ldots, n, \quad \forall t \in \mathcal{T}_B, \quad \forall m \in A_L(t), \tag{9}$$

$$\boldsymbol{a}_m^\top \boldsymbol{x}_i \geq b_t - M_2(1 - z_{it}), \quad i = 1, \ldots, n, \quad \forall t \in \mathcal{T}_B, \quad \forall m \in A_R(t), \tag{10}$$

Optimal Classification Tree

4. **Objective.** We suppose to solve a classification problem with K classes, and we use two auxiliary variables to model with Y_{ik} the misclassification of points and with N_{kt} the number of point of each class in a leaf. In addition, we define N_t to be the total number of points in a leaf node.

$$Y_{ik} = \begin{cases} +1, & \text{if } y_i = k \\ -1, & \text{otherwise} \end{cases}, \quad k = 1, \dots, K, \quad i = 1, \dots, n.$$

$$N_{kt} = \frac{1}{2} \sum_{i=1}^{n} (1 + Y_{ik}) z_{it}, \quad k = 1, \dots, K, \quad t \in \mathcal{T}_L, \tag{15}$$

$$L_t = N_t - \max_{k=1,\dots,K} \{N_{kt}\} = \min_{k=1,\dots,K} \{N_t - N_{kt}\}, \tag{19}$$

$$\min \quad \frac{1}{\hat{L}} \sum_{t \in \mathcal{T}_L} L_t + \alpha \sum_{t \in \mathcal{T}_B} d_t. \tag{23}$$

Empirical Decision Model Learning

Empirical Model Learning [Ref. 3] is a technique to enable decision making over complex real-world systems.
The idea is to:

1. Use Machine Learning to approximate the input/output of a system that is hard to model by conventional means.
2. Embed such *Empirical Model* into a Combinatorial Optimization model.

The range of potential applications may include:

- Applying Combinatorial Optimization to **Complex Systems** (in the proper sense), or systems that are too complicated to obtain by expert-design.
- Enabling ***Prescriptive*** analytics by taking advantage of a pre-extracted *predictive* analytics model.
- **Self adapting** systems, that could be obtained by retraining the Empirical Model.
- …

A *predictive* model describes the future evolution of the system/process dynamic. A *prescriptive* model provides decision support and allows to assess the effects of decisions on the system/process evolution in the medium or long term.

Empirical Decision Model Learning

Native Constraint Language:

$$\begin{aligned} \min \quad & x_0 \\ s.t. \quad & \pi_i(\vec{x}) \qquad \forall i \in I \\ & \vec{x} \in D_{\vec{x}} \end{aligned}$$

Where $\vec{x}$ is the vector of problem variables and $D_{\vec{x}}$ their domain.
The set I contains the indices of all constraints, represented here as predicates $\pi_i(\vec{x})$ that must hold in any feasible solution.
The x_0 represents the cost to be minimized.

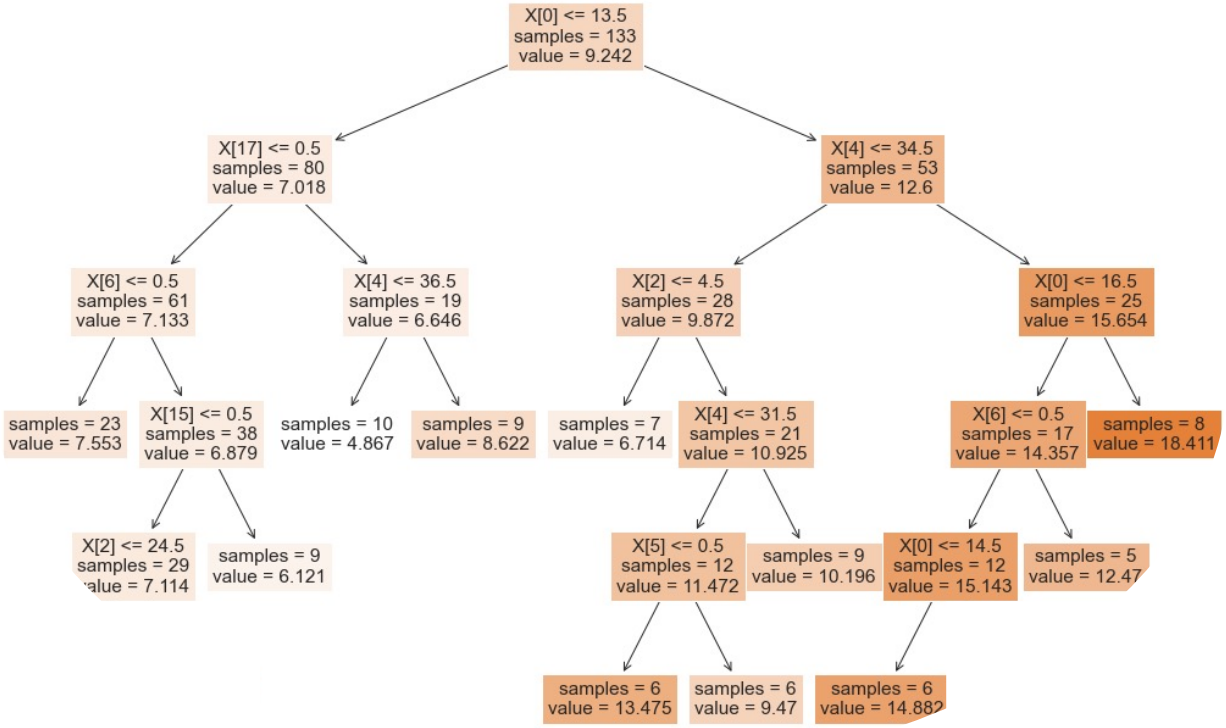
Incorporating Machine Learning Models:

$$\begin{aligned} \min \quad & x_0 \\ s.t. \quad & \pi_i(\vec{x}) \qquad \forall i \in I \\ & \vec{x}_{m,\,out} = m(\vec{x}_{m,\,in}) \qquad \forall m \in M \\ & \vec{x} \in D_{\vec{x}} \end{aligned}$$

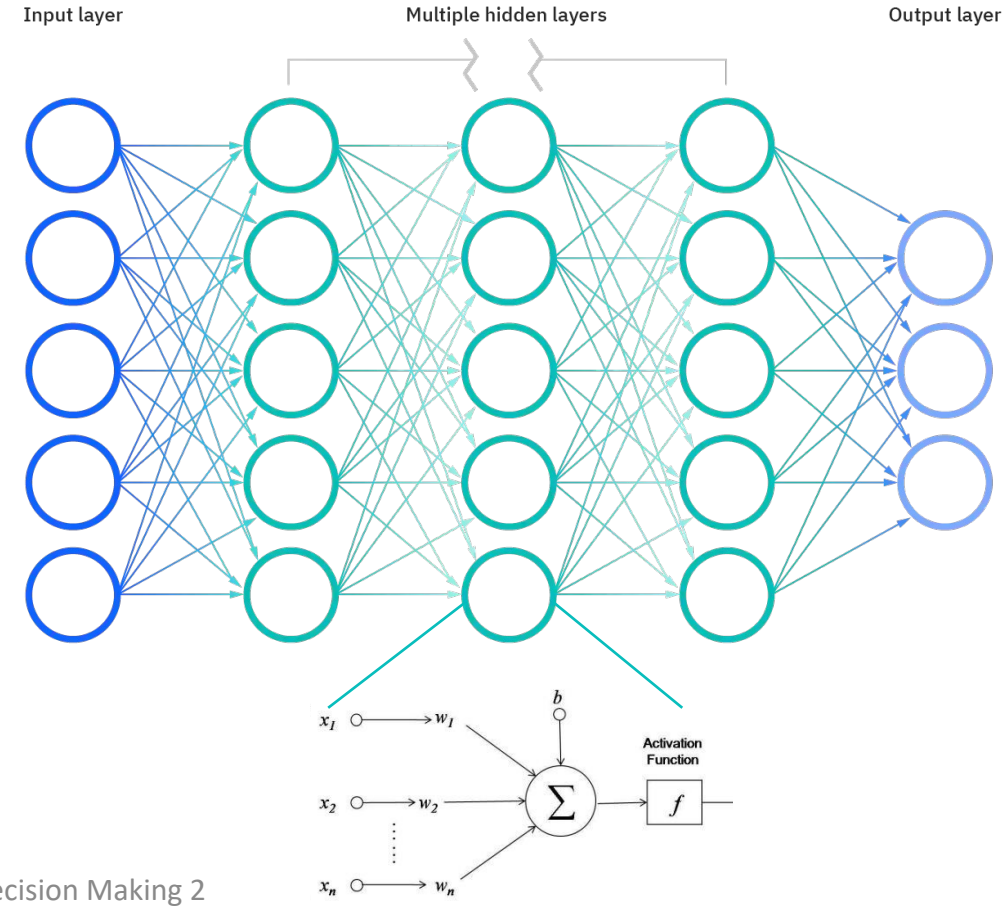
Where the new predicate is a machine learning model from a set M that is satisfied iff the output variable matches the evaluation of the model.

Empirical Decision Model Learning

Decision Tree



FeedForward Neural Network



References

1. Molnar, Christoph. ”*Interpretable machine learning”*. 2020.
2. Messalas, Andreas, Yiannis Kanellopoulos, and Christos Makris. "Model-agnostic interpretability with shapley values." *2019 10th International Conference on Information, Intelligence, Systems and Applications*
3. Bertsimas, Dimitris, and Jack Dunn. "*Optimal classification trees*”. In *Machine Learning.*
4. Lombardi, Michele, and Michela Milano. "*Boosting combinatorial problem modeling with machine learning*."