

# Introduction au machine learning

- **caractéristiques** : informations connues (ex : couleur, taux d'alcool, ...)
- **classe** : information qu'on souhaite prédire (ex : bière ou alcool ?)
- **données d'apprentissage** : données utilisées pour apprendre
- **données de test** : données utilisées pour évaluer les performances
- **précision** : % de données de test correctement classifiées
- **rappel** : % de données d'apprentissage correctement classifiées

## 1 Génération de règles

- $d$  : nombre de caractéristiques
- $t \in \{0, 1\}^d$  : transaction représentant une donnée
- $n$  : nombre de transactions dans les données d'apprentissage
- $b \in \{0, 1\}^d$  : règle
- $b \leq t$  :  $b$  s'applique à  $t$
- $y$  : classe (d'une transaction ou d'une règle)
- $S$  : ensemble des transactions de classe  $y$
- $x_i$  : la règle s'applique à  $t_i$
- couverture :  $s_X = \frac{1}{n} \sum_{i=1}^n x_i$
- support :  $s = \frac{1}{n} \sum_{i \in S} x_i$
- $R_{genX}, R_{genB}$  : coefficients de l'objectif
- $\bar{s}_X$  : borne supérieure sur la couverture
- $mincov$  : valeur minimale de  $\bar{s}_X$
- $P_{\bar{s}_X}$  : problème de génération d'une règle dont la couverture est  $\leq \bar{s}_X$
- $iter\_lim$  : nombre maximum de règles générées pour une valeur de  $\bar{s}_X$  donnée
- $\bar{s}$  : valeur de l'objectif du dernier problème  $P_{\bar{s}_X}$  résolu
- $\mathcal{R}_Y$  : ensemble des règles générées

## 2 Classement des règles générées

- $L$  : nombre de règles générées
- $u_{il}$  : la règle  $l$  est la plus haute s'appliquant à  $t_i$
- $p_{il} = \begin{cases} 0 & \text{si la règle } l \text{ ne s'applique pas à } t_i \\ 1 & \text{si la règle } l \text{ classe correctement } t_i \\ -1 & \text{si la règle } l \text{ ne classe pas correctement } t_i \end{cases}$
- $v_{il} = |p_{il}|$
- $r_l$  : rang de la règle  $l \forall l \in \{1, \dots, L\}$
- $r_*$  : rang de la plus haute règle nulle
- $R_{rank}$  : coefficient de l'objectif
- $g_i$  : rang de la plus haute règle satisfaisant  $t_i$
- $s_{lk}$  : la règle  $l$  a le rang  $k$
- $r_A$  : rang de la règle  $\emptyset \Rightarrow -1$
- $r_B$  : rang de la règle  $\emptyset \Rightarrow 1$
- $\alpha$  :  $r_*$  est égal à  $r_b$
- $\beta$  :  $r_*$  est égal à  $r_a$

## 3 Julia

### 3.1 Déclaration

- # Entier  
n = 10
- # Chaîne de caractères  
b = "Hello world"
- # Vecteur  
v = [1 2 3 4]
- # Matrice 2x2  
m = [1 2; 3 4]
- # Vecteur de 1 à n  
v = 1:n

### 3.2 Affichage et écriture

- # Inclure un fichier  
include("monFichier.dat")
- # Affichage console  
println("Afficher une variable", a)
- # Fichier en écriture  
fout = open("monFichierDeSortie.dat", "a")
- # Écrire dans ce fichier  
print(fout, v)

### 3.3 Structures de contrôle

- # Conditionnelle  
if v[1] == 1  
| # Contenu du if  
else  
| # Contenu du else  
end
- # Boucle for  
for i in 1:10  
| print(i)  
end
- # Boucle while  
while(v[1] == 1)  
| # Contenu de la boucle  
end

## 3.4 Optimisation

### 3.4.1 Modèle

- # Déclaration d'un problème d'optimisation avec CPLEX  
using JuMP  
using CPLEX  
m = Model(solver = CplexSolver())

### 3.4.2 Variables

- # Variable continue  
@variable(m, 0 <= x1 <= 1)
- # Variable binaire  
@variable(m, x2, Bin)
- # Tableau de taille n\*4  
@variable(m, 0 <= t[i in 1:n, j in 1:4] <= 1)

### 3.4.3 Contraintes

- #  $x_1 + x_2 = 1$   
@constraint(m, nomFacultatifDeLaContrainte, x1 + x2 == 1)
- #  $t_{ij} + x_1 \geq 1 \quad \forall i \in \{1, \dots, n\} \quad \forall j \in \{1, \dots, 4\}$   
@constraint(m, [i = 1:n, j = 1:4], t[i, j] + x1 >= 1)
- #  $\sum_{i=1}^n y_i \geq 3$   
@constraint(m, y[i] >= 3 for i in 1:n)

### 3.4.4 Objectif et résolution

- # Objectif  
@objective(m, Max, sum(y[i] for i = 1:n if v[i] == 2))
- # Résoudre le problème  
solve(m)
- # Valeur entière d'une variable  
vx1Int = round(Int, getvalue(x1))

### 3.4.5 Autre

- # Modifier le second membre d'une contrainte  
JuMP.setRHS(nomDeLaContrainte, 2)
- # Masquer les sorties de CPLEX  
m = Model(solver=CplexSolver(CPX\_PARAM\_SCRIND=0))