Mesure d'évaluation, Multi-classes

Cours 7
ARF Master DAC

Nicolas Baskiotis
nicolas.baskiotis@lip6.fr
http://webia.lip6.fr/~baskiotisn

équipe MLIA, Laboratoire d'Informatique de Paris 6 (LIP6)
Université Pierre et Marie Curie (UPMC)

S2 (2016-2017)

Plan

Mesures d'évaluation

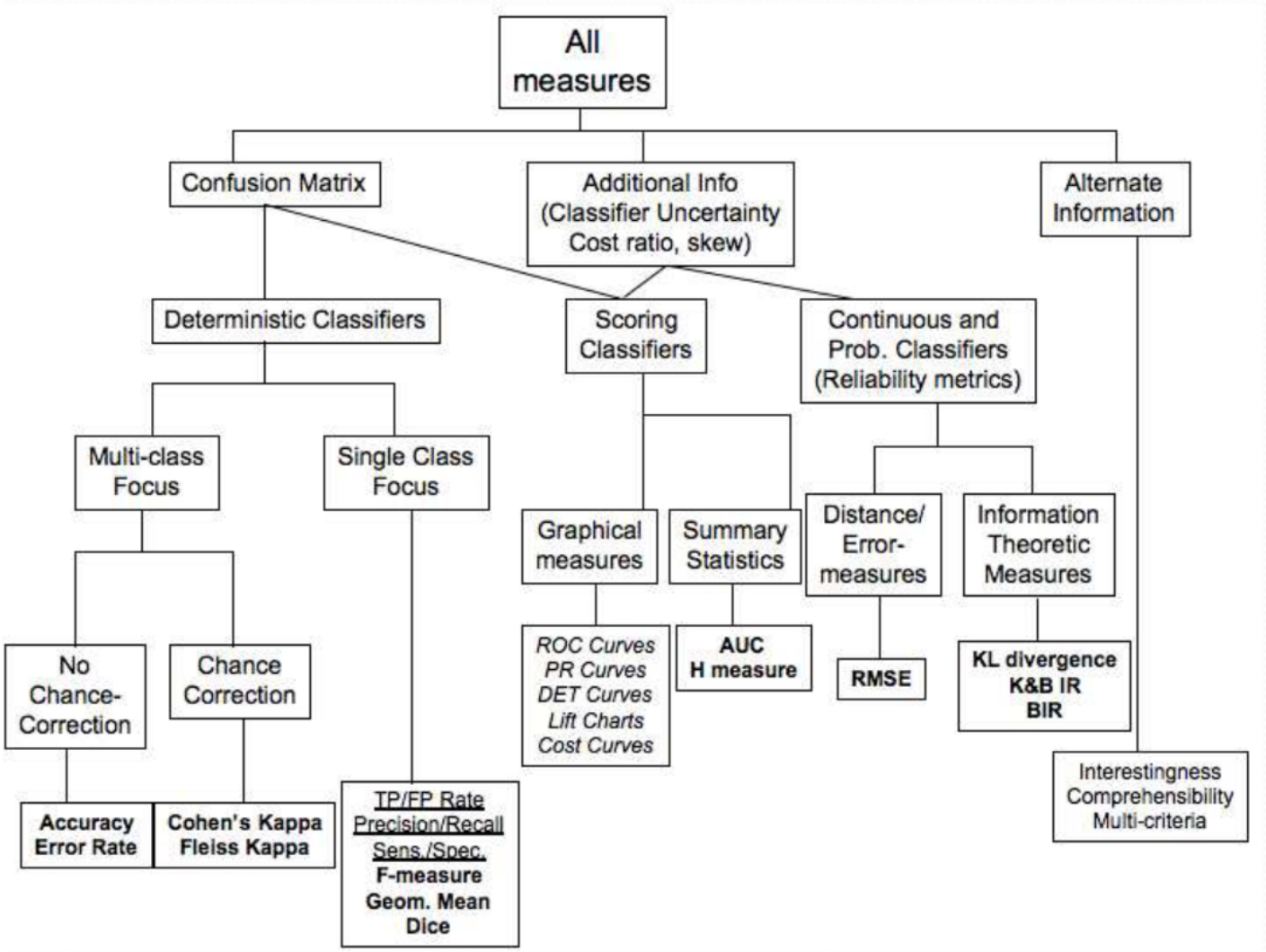
Objectifs

- Estimer la qualité des prédictions fournies par une approche
- Comparer des approches entre elles sur un problème donné
- Comparer des algorithmes sur un ensemble de problèmes

Le résultat dépend

- Choix de la mesure
- Choix du protocole de test (paramétrisation)
- Choix de l'échantillage

Une mesure unique ?



Tutorial icmla 2011, N. Japkowicz

Matrice de confusion

Contexte

- Un problème de classification binaire, étiquettes positif/négatif
- TP : Vrai positif (*True positive*), TN : Vrai négatif (*True negative*)
- FP : Faux positif (*False positive*), FN : Faux négatif (*False negative*)

Matrice de confusion

	Label $+$	Label $-$
$f(x) = +1$	TP	FP
$f(x) = -1$	FN	TN
	$P = TP + FN$	$N = FP + TN$

Mesures dérivées

- Erreur $0-1$: $\frac{FP+FN}{P+N}$
- Précision : $\frac{TP}{TP+FP}$
- Rappel (TP rate) : $\frac{TP}{P}$
- FP Rate : $\frac{FP}{N}$
- $F_\beta = (1+\beta^2)\frac{\text{precision}\times\text{rappel}}{\beta^2\text{precision}+\text{rappel}}$

Exemple (ou le problème du déséquilibre)

	Label +	Label −
$f(x) = +1$	200	100
$f(x) = -1$	300	400
	500	500

- Erreur : 60%
- Précision : 40%, Rappel : 40%
- F_1 : 0.4

	Label +	Label −
$f(x) = +1$	400	300
$f(x) = -1$	100	200
	500	500

- Erreur : 60%
- Précision : 66%, Rappel : 80%
- F_1 : 0.66

	Label +	Label −
$f(x) = +1$	200	100
$f(x) = -1$	300	400
	500	500

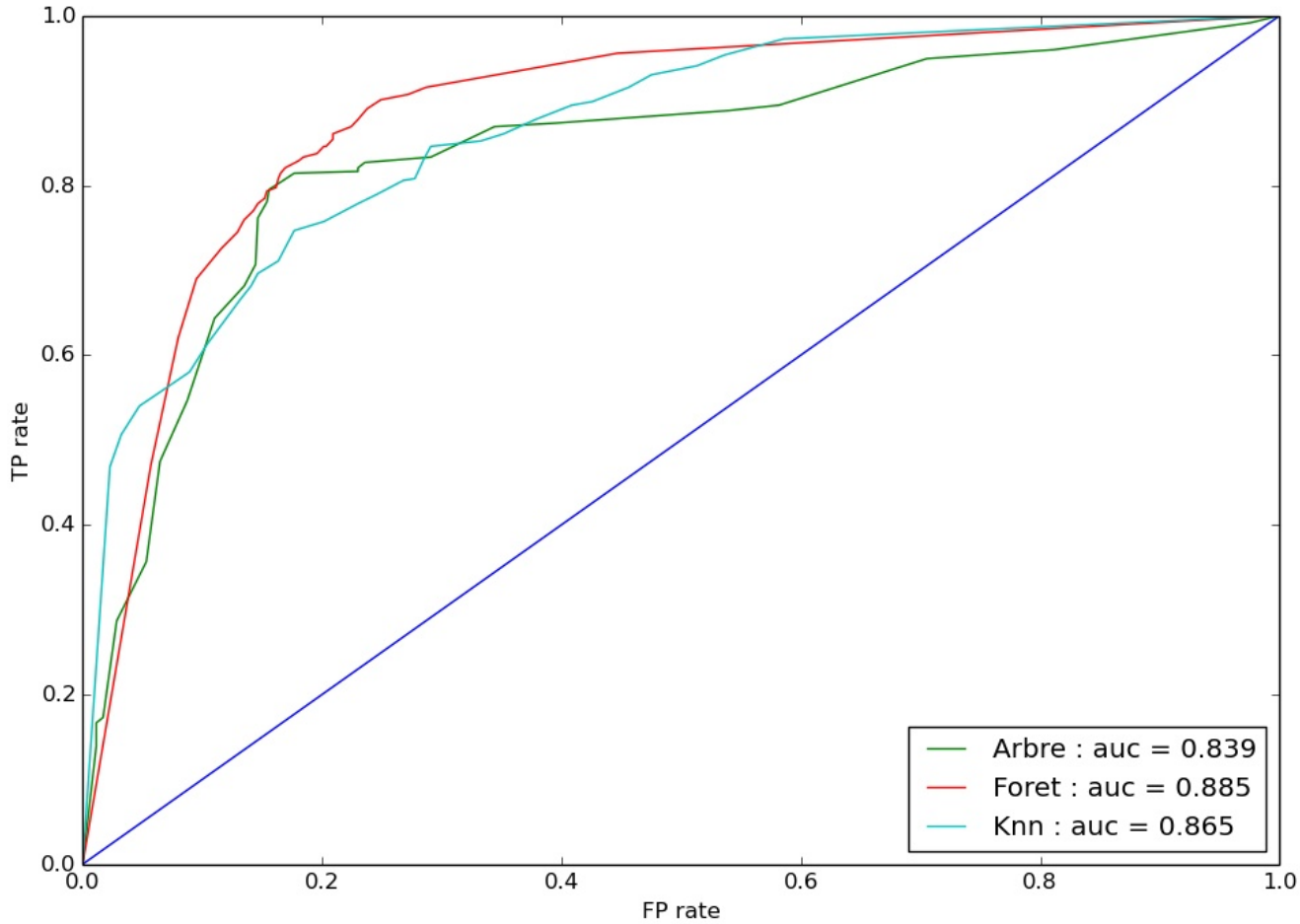
- Erreur : 60%
- Précision : 66%, Rappel : 40%
- F_1 : 0.5

	Label +	Label −
$f(x) = +1$	200	100
$f(x) = -1$	300	0
	500	100

- Erreur : 66%
- Précision : 66%, Rappel : 40%
- F_1 : 0.5

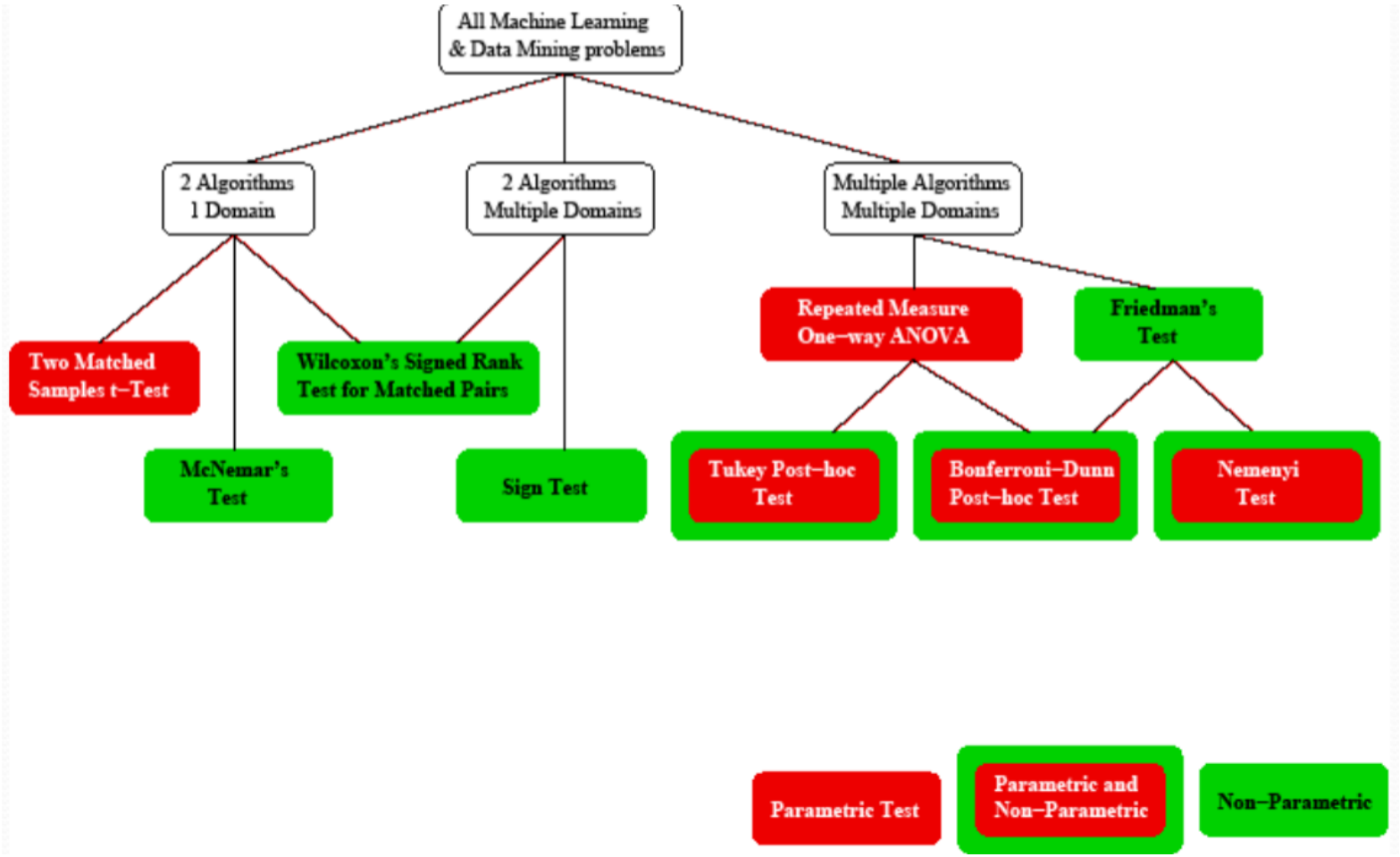
Courbe ROC et AUC

- Courbe ROC : TP rate en fonction du FP rate
- permet de calibrer un classifieur
- mesure d'intérêt : AUC, aire sous la courbe



Comment comparer deux algos ?

- Test statistique



Plan

Cas usuel

Contexte

- Classes : $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$
- Classification binaire ne marche pas directement

Approches “naïves” utilisant la classification binaire

- One-versus-one : matrice $M_{ij} = C_i$ vs C_j
- One-versus-all : vecteur $M_i = C_i$ vs $\{C_{j \neq i}\}$

Adaptation de la classification binaire

- Arbres, forets, k-nn : adaptation triviale
- SVMs multi-classes
- Réseau de neuronnes : vecteur de sortie $\mathbf{y}$ et softmax : $p(y_j) = \frac{e^{-y_j}}{\sum_i e^{-y_i}}$

Très grand nombre de classes

Problèmes des approches usuelles

- Coût d'une classification τ
- au mieux linéaire en fonction de K : temps τK
- grand nombre de dimensions

$\Rightarrow$ passage à l'échelle difficile en temps de calcul et en perfs

Deux grandes familles d'approche

- Approche *flat* : plonger les classes dans un espace $\mathbb{R}^{K'}$, $K' << K$
  Intérêt : $K'\tau$ pour trouver la bonne classe
- Approche *hiérarchique* : organiser les classes hiérarchiquement dans un arbre de classes
  Intérêt : inférence en $log(K)\tau$ pour un arbre binaire

Approches Error Correcting Output Code (ECOC)

Principe

- Plonger les classes dans $\mathbb{R}^{K'}$, $K' << K$
- Codage : une classe $\Leftrightarrow$ un code dans K'
- Inférence $=$ codage : $f : X \rightarrow K'$, $f(x)$ donne un code dans K'
- Décodage : classe dont le code est le plus proche

En pratique

- Un code c^i : un vecteur ternaire de K : $(-1, 0, 1, \ldots, 0, 1)$
- A chaque code, un classifieur binaire f_i qui sépare $\{C_j | c^i_j > 0\}$ et $\{C_j | c^i_j < 0\}$
- Matrice M de codage de K' : matrice $K' \times K$ des $M_{ij} = c^i_j$
- Codage d'une classe C_j : $(c^1_j, c^2_j, \ldots c^{K'}_j)$
- codage d'un exemple : $(f_1(x), f_2(x), \ldots, f_{k'}(x))$
- Inférence : $argmin_j d(f(x), M_{.j})$ en $O(K'\tau + K)$

Approche hiérarchique

Objectif

- Construire un arbre de partitionnement (hard ou soft) des classes
- Pour un nœud n :
  - un ensemble $\mathcal{C}_n$ de classes, pour les fils $n_1, \ldots, n_c$ sous-ensembles $\mathcal{C}'_{n_1}, \ldots, \mathcal{C}'_{n_c} \subset \mathcal{C}_n$, et $\bigcup \mathcal{C}'_{n_j} = \mathcal{C}_n$
  - un classifieur f_n à valeur dans $\{n_1, \ldots, n_c\}$
- Racine : ensemble de toutes les classes, feuilles : une seule classe
- Classification : un chemin dans l'arbre (en utilisant f_n), classe de la feuille

Problèmatiques

- Construire l'hiérarchie :
  - information a priori sur les classes : ontologie ou hiérarchie des classes
  - apprentissage de l'hiérarchie : clustering, approches gloutonnes
- Apprendre les classifieurs : problème de données non équilibrés
- Correction des erreurs : redondance des classes dans les nœuds de l'arbre