IA et science des données

Cours 10 – mardi 1er avril 2025

Clustering (fin). Retour au supervisé.

Christophe Marsala

Sorbonne Université

LU3IN026 - 2024-2025

Plan du cours

Apprentissage non-supervisé
- l'algorithme des K-moyennes (ou K-means)
- en pratique
- évaluation du résultat

Retour au supervisé : méthodes d'ensembles

Algorithme K moyennes (rappel)

- Prérequis
  - X : un ensemble de données (base d'apprentissage)
  - un entier naturel $K > 0$ (le nombre de clusters à trouver)
  - une mesure de distance d entre deux exemples x et y : $d(x, y)$
- Algorithme :
  1. choisir aléatoirement K exemples dans X comme premiers centres de clusters c_1, c_2,..., c_K
     - chaque centre c_k définit un cluster C_k
  2. affecter chaque x de X au cluster dont il est le plus proche
     - calculer $d(x, c_1)$,..., $d(x, c_K)$
     - affecter x au cluster C_k pour lequel $d(x, c_k)$ est la plus petite
  3. mettre à jour les centres des clusters
     - c_k est la moyenne des descriptions du cluster C_k
  4. retourner à l'étape 2 jusqu'à ce que l'inertie globale ne change plus beaucoup
- Résultat
  - un ensemble de clusters C_1,..., C_K

Évaluation du résultat d'un clustering

- Évaluer la partition obtenue : mesurer sa qualité
  - différentes approches
  - utilisation des caractéristiques des clusters
- Compacité d'un cluster
  - évaluer combien les exemples sont proches les uns des autres
  - compacité intra-cluster
- Séparabilité des clusters
  - évalue combien les clusters sont éloignés les uns des autres
  - distance inter-clusters
- Mesure globale : index d'une partition *(→ tableau)*
  - index de Dunn
  - index de Xie-Beni
  - ...

Plan du cours

Apprentissage non-supervisé

Retour au supervisé : méthodes d'ensembles

Biais et Variance (1)

- Apprentissage : trouver f, fonction de prédiction, telle que :

$$y = f(\mathbf{x}) + \epsilon$$

  avec $\epsilon \geq 0$ le plus petit possible
- idéalement : $\epsilon = 0$ (mais on n'y arrive jamais...)
- la "forme" de f est importante : elle utilise les variables de $\mathbf{x}$
  - linéaire, quadratique,...
  - arbre de décision
  - ...
- Modèle parcimonieux : nombre réduit de variables utilisées,...
  - idée : modèle parcimonieux $\Longrightarrow$ faible variance
- Biais : complexité du modèle
- Variance : capacité du modèle à changer si la base d'apprentissage change

Biais et Variance (2)

- Objectif : faible biais & variance faible
  - très difficile d'atteindre les 2... il faut choisir !
- Nouvelle approche : réduire la variance
  - combiner plusieurs classifieurs : ensemble de classifieurs
  - agréger leur résultats pour améliorer les performances
- Différentes façons de faire
  - on regarde avec les arbres (par exemple)
  - multiplier les arbres pour les combiner ensuite

L'approche BAGGING

- Bootstrap AGGregatING
- Construire un ensemble de classifieurs de même type
- Agréger leurs résultats lors d'une classification
- $\Longrightarrow$ approche très efficace !
  - la variance globale est plus faible que la variance de chaque classifieur
- Si les classifieurs sont des arbres de décision : forêt

L'approche BAGGING : apprentissage et classification

Apprentissage :

- Soit $\mathbf{X}$ une base d'apprentissage avec n exemples
- Soit B le nombre de classifieurs souhaités
  1. Extraire B sous-bases de $\mathbf{X}$: $\mathbf{X}_1, \ldots, \mathbf{X}_B$
  2. Construire un classifieur f_k pour chaque sous-base $\mathbf{X}_k$
- Au final : on obtient un ensemble de B classifieurs $f_1, \ldots, f_B$

Classification :

- Soit un ensemble de B classifieurs $f_1, \ldots, f_B$
- Soit un exemple $\mathbf{x}$ à classer

Les forêts aléatoires (random forest)

- Idée : plus les arbres sont diversifiés, meilleur sera le score global
  - augmenter la diversité : plus d'aléatoire !
  - $\rightarrow$ random forest
- Soit $\mathbf{X}$ une base d'apprentissage avec n exemples
- Soit B le nombre de classifieurs souhaités, $m < n$ le nombre d'exemples à choisir et $p \leq d$ variables de description à choisir
  1. Extraire B sous-bases de $\mathbf{X}$: $\mathbf{X}_1, \ldots, \mathbf{X}_B$
  2. Construire un classifieur f_k pour chaque sous-base $\mathbf{X}_k$
- Remarque : B, m et p sont des hyper-paramètres de l'algorithme