ALGORITHMES POUR L'ÉTIQUETAGE ET LA CLASSIFICATION

Sommaire

Nous allons, exposer des méthodes de classement et de classification automatique. Nous commençons par en préciser la problématique et les notions sous-jacentes.

I Classement et classification automatique

Définition 7.1: Partition

Soit E un ensemble. I et J des ensembles quelconques

- Une partition de E est la donnée d'une famille $(A_i)_{i\in I}$ de parties ou sous-ensembles non-vides de E, telle que : $\begin{cases} \bigcup_{i\in I} A_i & = E \\ A_i \cap A_j & = \emptyset, \text{ pour tous } i \neq j \end{cases}$
- Soient $(A_i)_{i\in I}$ et $(B_j)_{j\in J}$ deux partitions du même ensemble E. On dit que $(B_j)_j$ est plus fine que $(A_j)_j$ ssi $\forall j \in J, \exists i \in I, B_j \subset A_i$.

Proposition 7.2

- Si R est une relation d'équivalence sur un ensemble E, ses classes d'équivalence forment une partition de E.

- Pour toute partition P de E il existe une relation d'équivalence sur E telle que P est l'ensemble des classes d'équivalence de R. Elle peut trivialement se définir par « $R(x, y)$ ssi x et y appartiennent à un même élément de P ».

Définition 7.3: Classer ou étiqueter

Classer, étiqueter les éléments d'un ensemble (documents écrits, images, individus, lieux géographiques, événements ou données quelconques) revient à placer ces éléments dans des classes qui forment une partition de cet ensemble. Dans une autre formulation, cela revient à associer à chaque élément une étiquette appartenant à une liste prédéfinie (les éléments d'une même classe se voyant affublés d'une même étiquette). Les propriétés des partitions assurent que l'opération de classement d'un élément e est toujours définie de façon unique.

Définition 7.4: Classifier

Classifier est l'opération qui vise à définir une partition ou un système de partitions pour un ensemble de données. On imagine sans peine qu'en pratique on visera à obtenir une classification dans laquelle les individus appartenant à une même classe seront le plus ressemblants/proches possibles, les individus appartenant à des classes distinctes devront être le moins ressemblants/proches possible. Cette notion de ressemblance ou de proximité dépendra du type des données et nous serons amenés à considérer des distances ou des mesures de similarité adaptées à chaque problème.

Exemple 7.5

- Classifier des mails en "spams" et "non-spams"
- Classifier les réels : les positifs et les négatifs stricts.
- Classifier des romans : "science-fiction", "romantique", etc..
- Classifier un ensemble de portraits selon si ce sont des portraits de chats, de chiens, d'humains, etc..

II Distance ou similarité

II.1 Distance

Définition 7.6: distance

Soit E un ensemble, une distance sur E est une application de $E \times E$ dans $\mathbb{R}_+$ telle que pour tout $(x, y, z) \in E^2$:

1. $d(x, x) = 0$ et $d(x, y) = 0 \Rightarrow x = y$ (axiome de séparation) ;
2. $d(x, y) = d(y, x)$ (symétrie) ;
3. $d(x, y) \leq d(x, z) + d(z, y)$ (inégalité triangulaire).

Exemple 7.7

On peut en particulier définir une distance sur une partie d'un espace vectoriel à partir d'une norme en posant $d(X,Y) = \mathcal{N}(X-Y)$. Ainsi, dans $\mathbb{R}^n$,

$$d(X,Y) = \|X-Y\|_1 = \sum_{i=1}^{n} |x_i - y_i|$$
$$d(X,Y) = \|X-Y\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$
$$d(X,Y) = \|X-Y\|_\infty = \max_{1\leq i\leq n} |x_i - y_i|$$

Exemple 7.8: Distance de Levenshtein

Soient $X = X_0$ et Y deux mots ou chaînes de caractères. On s'autorise les opérations élémentaires $X_p \longrightarrow X_{p+1}$ suivantes :

- X_{p+1} est obtenue à partir de X_p par suppression d'un caractère ;
- X_{p+1} est obtenue à partir de X_p par insertion d'un caractère de Y ;
- X_{p+1} est obtenue à partir de X_p par remplacement d'un caractère par un caractère différent provenant de Y.

La distance de Levenshtein entre X et Y est égale à la longueur minimale d'une suite d'opérations du type de celles précédemment décrites (on dira de types S, I_Y ou R_Y) qui permet de transformer X en Y. On la note $\mathrm{Lev}(X,Y)$.

On peut démontrer que Lev est bien une distance.

II.2 Mesures de similarité

Les distances usuelles dans les problèmes numériques ou qui ont une modélisation géométrique ne sont pas adaptées à toutes les situations. Nous ferons alors intervenir d'autres distances ou, à défaut, des mesures ou indices de similarité dont nous donnons ici quelques exemples.

Exemple 7.9: Indice et distance de Jaccard

On définit a priori l'indice de Jaccard comme un mesure de similarité entre deux ensembles finis A et B (l'un d'eux étant non vide) en posant

$$J(A,B) = \frac{|A\cap B|}{|A\cup B|} \text{ qui a les propriétés } \begin{cases} 0 \leq J(A,B) \leq 1 \\ J(A,B) = 1 \Leftrightarrow A = B \\ J(A,B) = 0 \Leftrightarrow A\cap B = \emptyset \end{cases}$$

Nous pouvons lui associer la distance de Jaccard, définie par

$$d_J(A,B) = 1 - J(A,B) = 1 - \frac{|A\cap B|}{|A\cup B|} = \frac{|A\Delta B|}{|A\cup B|}.$$

où $A\Delta B$ désigne la différence symétrique entre A et B : $A\Delta B = (A\cup B)\backslash(A\cap B)$. Cet indice intervient lorsque nous voulons comparer des éléments caractérisés par des attributs booléens/binaires (présence ou non de certaines propriétés). Supposons que nous voulions comparer des individus selon qu'ils possèdent ou non des propriétés P_i. Chaque individu est

représenté par un vecteur de $\{0,1\}^n$ avec $x_i = 1$ ssi $P_i(x)$:

$$X = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \leftarrow \text{ ne vérifie pas } P_2, \quad Y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \leftarrow \text{ vérifie } P_2.$$

La distance de Jaccard est nulle lorsque les vecteurs X et Y sont identiques, égale à 1 lorsque $\forall i, x_i \neq y_i$.

Exemple 7.10: Comparaisons de textes, matrices termes-documents

Pour évaluer la proximité entre documents textes issus d'un même corpus on commence par en donner une représentation sommaire en repérant les mots utiles du corpus et en associant à chaque document D_j un vecteur colonne X_j dont le terme $x_{i,j}$ dépend de la fréquence du mot ou du terme t_i dans le document et dans le corpus $\mathcal{C}$. Le nombre d'occurrences du terme pourrait servir à comparer grossièrement des documents de même taille.

Cependant, on préfère le plus souvent le codage tf-idf (tf : fréquence ou plutôt nombre d'occurrences des termes dans les documents, idf : inverse du nombre de documents contenant un terme) défini par les formules

$$x_{i,j} = f_{t_i,d_j} \times \ln\left(\frac{|\mathcal{C}|}{df_{t_i}}\right) \text{ avec } \begin{cases} f_{t_i,d_j} & : \text{ fréquence de } t_i \text{ dans } D_j \\ df_{t_i} & : \text{ nombre de docs. contenant } t_i \end{cases}$$

Un indice de similarité entre deux documents est alors donné par la valeur du cosinus de "l'angle" entre X_j et X_k comme vecteurs de $\mathbb{R}^n$ avec n taille du lexique retenu (de quelques milliers à quelques dizaines de milliers de termes en pratique) :

$$\text{sim}\left(D_j, D_k\right) = \frac{\sum x_{i,j}x_{i,k}}{\left\|X_j\right\|_2 \left\|X_k\right\|_2} = \frac{\left(X_j \mid X_k\right)}{\left\|X_j\right\|_2 \left\|X_k\right\|_2}$$

où $\left(X_j \mid X_k\right)$ désigne le produit scalaire canonique dans $\mathbb{R}^n$. L'exercice qui suit permet de vérifier que deux documents similaires ont un indice de similitude proche de 1 , que deux documents très différents ont un indice de similitude proche de 0.

Exercice 7.11

On suppose que les documents d'un corpus sont représentés comme par des vecteurs X_j tels que $x_{i,j} = f_{t_i,d_j} \times \ln \frac{|\mathcal{C}|}{df_{t_i}}$.

1. On suppose que le mot ou le terme t_i est présent dans tous les documents. Que vaut alors $x_{i,j}$?
2. On suppose que le mot ou le terme t_i est présent dans le seul document D_j. Que vaut alors $x_{i,j}$?
3. On pose $\delta\left(D_j, D_k\right) = 1 - \text{sim}\left(D_j, D_k\right)$. Que dire des documents D_j et D_k tels que $\delta\left(D_j, Dk\right) = 0$? (Penser au cas d'égalité de Cauchy-Schwarz). S'agit-il d'une distance ?
4. δ vérifie-t-elle l'inégalité triangulaire ? Indication : on pourra réécrire une relation $\delta(X,Z) \leq \delta(X,Y) + \delta(Y,Z)$ en introduisant les notations

$$\cos\alpha = \frac{(X \mid Y)}{\|X\|_2, \|Y\|_2}, \cos\beta = \frac{(Y \mid Z)}{\|Y\|_2, \|Z\|_2}, \cos\gamma = \ldots$$

puis s'aider d'un dessin (on se souviendra que trois vecteurs de $\mathbb{R}_+^n$ sont coplanaires).

E

5. Que peut on dire de deux documents pour lesquels $\delta\left(D_j, D_k\right) = 1$?

III Inertie d'une partition

On définit l'inertie d'une partie finie de $\mathbb{R}^n$, puis d'une partition d'un ensemble, notion essentielle dans l'évaluation des algorithmes de classification.

Définition 7.12: Inertie

- Si $A = \left(X_j\right)_{0\leq j\leq m-1}$ est une partie finie de $\mathbb{R}^n$, le barycentre et l'inertie de A sont définis par :

$$G_A = \frac{1}{m}\sum_{j=0}^{m-1} X_j$$

$$I_A = \sum_{j=0}^{m-1} \left\|X_j - G_A\right\|_2^2$$

On appelle aussi variance de A, la moyenne $\frac{I_A}{|A|}$.

- Soit $(A_k)_{0\leq k\leq K-1}$ une partition d'un ensemble fini E contenu dans $\mathbb{R}^n$, on appelle inertie totale de la partition la somme des inerties de chacune de ses classes :

$$I = \sum_{k=0}^{K-1} I_{A_k} = \sum_{k=0}^{K-1}\sum_{X_j\in A_k} \left\|X_j - G_{A_k}\right\|^2$$

Remarque 7.13. *Comme une variance en mathématique, plus la variance est petite, plus les points sont proches du barycentre, cela revient à dire ici que les éléments (d'une même partition) se "ressembleront". Le but sera donc d'essayer de rendre l'inertie totale la plus faible possible.*

IV Intelligence artificielle

L'intelligence artificielle (IA) est un terme souvent utilisé de manière large, englobant divers concepts, parfois de façon quelque peu floue, surtout lorsqu'il est abordé par des journalistes. Dans l'ensemble, on peut définir l'IA comme un "ensemble de théories et de techniques mises en œuvre en vue de réaliser des programmes informatiques qui visent à 'simuler' l'intelligence humaine". Bien que cela puisse être interprété de différentes manières, il existe un consensus général sur l'inclusion d'algorithmes, de méthodes, ou de programmes spécifiques dans le domaine, en fonction soit du problème à résoudre, soit des techniques mises en œuvre.

Certains problèmes abordés par l'IA sont souvent caractérisés par leur complexité, rendant difficile l'application d'une méthode exacte de bout en bout, ou parfois, ces problèmes ne se prêtent tout simplement pas à une formulation précise.

Caractérisation par les problèmes

- traitement du langage : utilisation d'algorithmes d'apprentissage automatique pour identifier la langue d'un texte en fonction de modèles linguistiques, diviser le texte en tokens

de manière "intelligente", analyser la structure grammaticale d'une phrase, trouver les sentiments exprimés dans le texte, les générateurs de texte en particulier les LLM (Large Language Model comme chatGPT).

- classement, étiquetage automatique (cf le début du chapitre) ;
- classification automatique, clustering (cf le début du chapitre) ;
- certains programmes de jeux dans lesquels interviennent des heuristiques...

Caractérisation par les méthodes

- systèmes experts : un outil capable de reproduire les mécanismes cognitifs d'un expert, dans un domaine particulier ;
- La programmation logique est une forme de programmation qui définit les applications à l'aide :
  - d'une base de faits : ensemble de faits élémentaires concernant le domaine visé par l'application,
  - d'une base de règles : règles de logique associant des conséquences plus ou moins directes à ces faits,
  - d'un moteur d'inférence (ou démonstrateur de théorème) : exploite ces faits et ces règles en réaction à une question ou requête.

  Cette approche se révèle beaucoup plus souple que la définition d'une succession d'instructions que l'ordinateur exécuterait.
- l'inférence bayésienne : est l'ensemble des techniques permettant d'induire les caractéristiques d'un groupe général à partir de celles d'un groupe particulier, en fournissant une mesure de la certitude de la prédiction, tout cela en utilisant principalement la formule de Bayes.
- apprentissage supervisé (cf la suite) ;
- apprentissage non supervisé (cf la suite) ;
- apprentissage profond (réseaux de neurones) ;

Cette diversité de caractérisations souligne la richesse et la variété des approches au sein du domaine de l'intelligence artificielle.

IV.1 Apprentissage supervisé

La problématique de l'apprentissage supervisé est la suivante :

- On dispose de données déjà classées ou étiquetées (s'il s'agit d'un problème de classement) ou de données sous la forme d'entrées-sorties. C'est l'ensemble d'apprentissage.
- On souhaite pour des données nouvelles du même type, attribuer une étiquette ou une valeur de sortie en se basant sur ce que l'on sait déjà grâce à l'ensemble d'apprentissage.

D'une façon générale, on se donne un modèle pour la fonction de prédiction f : entrées $\longrightarrow$ sorties, et on cherche à approcher f à partir des couples d'entrées-sorties fournis par l'ensemble d'apprentissage. Les différentes méthodes reposent sur des raisonnements probabilistes ou statistiques.

Exemple 7.14

- Reconnaissance de forme ;
- Reconnaissance vocale ;
- vision par ordinateur (transformation d'images visuelles en descriptions du monde qui ont un sens pour les processus de pensée et peuvent susciter une action appropriée, par exemple pour qu'un rover puisse se déplacer sur Mars).

IV.2 Apprentissage non supervisé

On parle d'apprentissage non supervisé lorsqu'il s'agit de découvrir les paramètres d'un modèle de structure sous-jacente à un ensemble de données en l'absence de connaissance a priori.

Exemple 7.15

- Partitionnement des données pour, par exemple, les compresser ou extraire des connaissances, pour faire émerger des sous-ensembles et sous-concepts éventuellement impossibles à distinguer naturellement.
- La réduction de la dimensionnalité (ou réduction de (la) dimension) : c'est un processus qui consiste à prendre des données dans un espace de grande dimension, et à les remplacer par des données dans un espace de plus petite dimension. Pour que l'opération soit utile il faut que les données en sortie représentent bien les données d'entrée.

V Classement supervisé, k-plus proches voisins

V.1 Définitions

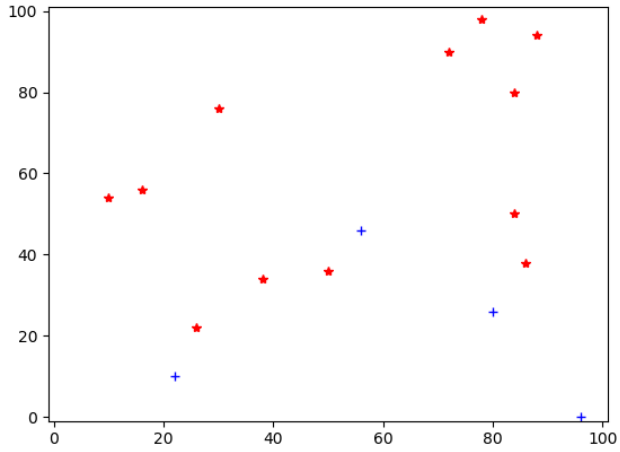
Théorème 7.16: Algorithme des k-plus proches voisins

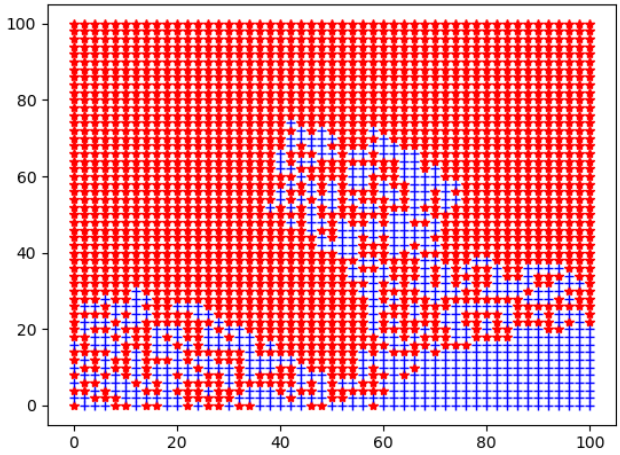
On veut classer ou étiqueter des données appartenant à un ensemble muni d'une distance ou d'une mesure de similarité δ. Les étiquettes sont déterminées et on dispose d'un ensemble d'apprentissage A dans lequel chaque donnée est étiquetée ou classée.

Algorithme 7.17: k plus proche voisins

L'idée de l'algorithme est la suivante : Pour prédire l'étiquette d'un élément X'

1. On calcule les k éléments de l'ensemble d'apprentissage qui sont les plus proches de X' (au sens de δ) ;
2. On attribue à X' l'étiquette la plus fréquente dans ce voisinage.





EXEMPLE D'UTILISATION DE L'ALGORITHME DES K-VOISINS VU EN TP AVEC LA DISTANCE EUCLIDIENNE.

V.2 Tests et matrice de confusion

En pratique on teste de façon systématique un algorithme supervisé. On procède en partant d'un ensemble de données étiquetées ou pour lesquelles les sorties sont connues, X. On sélectionne un sous-ensemble des données qui sera l'ensemble d'apprentissage, $A \subset X$. On compare alors les prédictions que fournit l'algorithme pour des éléments $x \in X \backslash A$ aux étiquettes connues. La matrice de confusion de ces deux étiquetages rassemble tous ces éléments de comparaison.

Définition 7.18: matrice de confusion

Soient X un ensemble de données, $\mathcal{L} = \{e_0, e_1, \ldots\}$ un ensemble fini d'étiquettes et f_1, f_2 deux applications $f_i : X \to \mathcal{L}$. La matrice de confusion de f_1, f_2 est la matrice M de taille $|\mathcal{L}| \times |\mathcal{L}|$ telle que $M[i,j] = \left|\{x \in X / f_1(x) = e_i, f_2(x) = e_j\}\right|$.

Remarque 7.19. *Si la matrice est diagonale, alors f_1 et f_2 étiquettent de la même manière. En pratique, si on a un jeu de tests, il sera important que la fonction f obtenue par l'algorithme ressemble à l'hypothétique et que la matrice de confusion des deux fonctions soit diagonale ou quasi-diagonale.*

VI Classification non supervisée, algorithme des k-moyennes

L'algorithme de classification que nous présentons ici est un algorithme non supervisé : aucune information préalable sur une partition, aucun exemple de partie ne sont fournis en donnée. Seul le nombre de classes dans la partition que l'on veut obtenir est précisé.

Théorème 7.20: Algorithme des k-moyennes

- On dispose de M documents notés $(D_0, D_1, \ldots, D_{M-1})$; ces documents peuvent être des textes, des images, des relevés statistiques provenant de mesures biologiques comme de comportements d'achats, etc.
- Chaque document est représenté par un vecteur $X_j \in \mathbb{R}^N$;
- On suppose que l'on dispose d'une fonction d'écart entre les documents dist (D_i, D_j) qui vérifie l'inégalité triangulaire.

Alors, la fonction : `k_moyennes(X,k)` prend en arguments un ensemble de documents $X = (X_j)_j$, un entier $k \geq 1$ et renvoie un tuple (P, C) où P est une liste de k parties formant une partition de l'ensemble des documents et C est la liste des k centres de gravité de ces parties.

Algorithme 7.21

- On se donne ou on choisit aléatoirement k documents dans $(D_j)_j$ pour initialiser la liste C. Ils sont notés $C_0, \ldots, C_{k-1}$.
- On réserve un ensemble P de k listes vides ;
- Pour chaque document représenté par X_j on calcule les k distances dist (D_j, C_ℓ) , $\ell = 0, \ldots, k-1$. Si la distance minimale est dist $(D_j, C_{\ell_{min}})$, on place D_j dans la classe $\ell_{\min}$ (on place donc j dans la liste $P[\ell_{\min}]$).
- On dispose d'une première partition, on calcule le centre de gravité de chaque classe. La liste C est mise à jour avec les k centres de gravité : $C = [C_0, \ldots C_{k-1}]$.
- On itère le processus jusqu'à ce que les classes ne soient plus modifiées ou en utilisant une autre propriété à définit selon l'utilisation.