Recommandation - Rapport partiel

Ken Chanseau–Saint-Germain & Vincent Vidal

20 janvier 2014

Table des matières

1 Introduction **2**

2 Approximation basique **2**

2.1 Par moyenne 2

2.2 Par minimisation de problème convexe 3

3 Approximation par voisininage **3**

3.1 Plus proche voisin 3

3.2 k plus proches voisins 4

3.3 Voisinage 4

4 Décomposition en valeurs singulières **5**

Notation

On notera pour $p > 0$ un réel, X un vecteur et M une matrice quelconque :

$$\|X\|_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}} \qquad \|M\|_p = \left(\sum_{i,j} |m_{i,j}|^p\right)^{\frac{1}{p}}$$

$$|||M|||_p = \sup_{\|x\|_p = 1} \|Mx\|_p$$

Et on posera $\|X\|_0$ le nombre de composantes non nulles de X.

1 Introduction

On se donne ici n personnes donnant des notes à m objets.
On notera $a_{i,j}$ la note de l'individu i sur l'objet j ainsi que $A = (a_{i,j})_{\substack{1\leq i\leq n\\1\leq j\leq m}}$ la matrice des notes.

On suppose ici que l'on a accès qu'à une matrice incomplète B obtenu en annulant certaines composantes de A. Le but est alors de trouver une bonne approximation $\widehat{A}$ de A à partir de B.

On prendra comme mesure d'approximation, l'erreur moyenne suivante :

$$\mathrm{RMSE}\left(\widehat{A}\right) = \left\|A - \widehat{A}\right\|_2 = \sqrt{\sum_{i,j} (a_{i,j} - \widehat{a}_{i,j})^2}$$

2 Approximation basique

2.1 Par moyenne

On suppose ici que la note d'un objet est de la forme $\widehat{a}_{i,j} = m_{\widehat{A}} + p_i + o_j$. Avec $m_{\widehat{A}}$ la moyenne des valeurs de $\widehat{A}$, p_i la moyenne des notes recentrées qu'a donné la personne i et o_j la moyenne des notes recentrées qu'a obtenu l'objet j. C'est à dire :

$$m_{\widehat{A}} = \frac{\sum_{i,j} b_{i,j}}{\|B\|_0} \qquad p_i = \frac{\sum_j b_{i,j} - m_{\widehat{A}}}{\|B^{\mathrm{T}}\mathrm{e}_i\|_0} \qquad o_j = \frac{\sum_i b_{i,j} - m_{\widehat{A}}}{\|B\mathrm{e}_i\|_0}$$

Algorithm 1: Recommandations naives par moyenne

Data: La matrice incomplète B.
Result: La matrice d'approximation $\widehat{A}$.
$m \longleftarrow \mathrm{mean}(B)$;
for $i \in [\![1,\, n]\!]$ **do**
  $p[i] \longleftarrow 0$;
  $card \longleftarrow 0$;
  for $j \in [\![1,\, m]\!]$ **do**
    $p[i] \longleftarrow p[i] + B[i,\, j]$;
    if $b[i,\, j] \neq 0$ **then**
      $card \longleftarrow card + 1$;
  $p[i] \longleftarrow p[i]/card - m$;
On calcul de même les $o[j]$;
for $i \in [\![1,\, n]\!]$ **do**
  for $j \in [\![1,\, m]\!]$ **do**
    $\widehat{A}[i,\, j] \longleftarrow m + p[i] + o[j]$;
return $\widehat{A}$;

2.2 Par minimisation de problème convexe

De manière plus précise, on peut résoudre le problème suivant de minimisation convexe pénalisé sur les p_i et o_j pour trouver des p_i et o_j plus adaptés :

$$\mathcal{P}\left(p,\ o\right) = \sum_{i,j} \left(a_{i,j} - m_{\widehat{A}} + p_i + o_j\right)^2 + \lambda \left(\left\|o\right\|_1 + \left\|p\right\|_1\right)$$

Problème que l'on peut écrire :

$$\mathcal{P}\left(x\right) = \left\|Mx + b\right\|_2^2 + \lambda \left\|x\right\|_1$$

Il suffit alors de résoudre ce problème, pour un λ donné et de calculer l'approximation à partir des coefficients p_i et o_j obtenus.

On utilisera ici un algorithme de minimisation présenté dans [?], se rapprochant d'une méthode de descente de gradient. Cet algorithme prend un paramètre supplémentaire, μ, dont dépend la vitesse de convergence. La suite calculée par cet algorithme converge vers un minimum lorsque μ est suffisamment proche de 0.

Algorithm 2: Minimisation convexe

Data: x, λ, le nombre d'itération N et un paramètre μ.
Result: une meilleur approximation x.
for $k \in [\![1,\ N]\!]$ **do**
    $x \longleftarrow x - \mu.M^{\mathrm{T}}\left(Mx - b\right)$;
    `/* Seuillage doux sur chaque composante */`
    for $i \in [\![1,\ n]\!]$ **do**
        $x\left[i\right] \longleftarrow \mathrm{signe}\left(x\left[i\right]\right).\max\left(\ \left|x\left[i\right]\right| - \mu\lambda,\ 0\ \right)$;
return x;

3 Approximation par voisininage

Nous noterons par la suite P_i la i-ième colonne de B^{T}, représentant la personne i et O_j la j-ième colonne de B, représentant l'objet j.

3.1 Plus proche voisin

La méthode précédente suppose qu'il n'existe qu'un seul type de personne et d'objet : un objet est bien ou mauvais dans l'ensemble et une personne donne de manière générale des notes bonnes ou des notes mauvaises. Ceci apparaît comme très naif.

On considère ici une méthode differente. Pour une personne i et un objet j dont l'information est manquante, nous allons chercher parmis les personnes ayant noté j celui qui se rapproche le plus i ainsi que l'objet ayant été noté par i celui qui se rapproche le plus de j. En notant i_0 et j_0 respectivement la personne et l'objet trouvé, on peut alors prendre comme approximation la moyenne des deux notes : $\widehat{a}_{i,j} = \frac{1}{2}\left(a_{i_0,j} + a_{i,j_0}\right)$.

Il reste à définir comment sélectionner le plus proche voisin. Si on cherche le plus proche voisin d'un vecteur X de support S parmis un ensemble de vecteur E, on considèrera $E_{|S}$ les vecteurs de E restreints au support S et on prendra alors le plus proche voisin pour la norme euclidienne sur $\mathbb{R}^{\mathrm{Card}(S)}$.

3.2 k plus proches voisins

3.3 Voisinage

On ne cherche plus ici le plus proche voisin, mais une combinaison de la note donnée par les personne ayant noté l'objet en question, et ce en fonction de leur ressemblance avec la personne initiale.

Il convient de donner une mesure de ressemblance entre deux personnes i_1 et i_2, que l'on notera $s\left(i_0,\ j_0\right)$. Nous prendrons ici la valeur du cosinus de l'angle entre les deux personnes :

$$s\left(i_1,\ i_2\right) = \max\left(\frac{P_{i_1}^{\mathrm{T}} \cdot P_{i_2}}{\left\|P_{i_1}\right\|_2 \left\|P_{i_2}\right\|_2},\ 0\right)$$

On prendra alors comme approximation :

$$\widehat{a}_{i,j} = \frac{\sum\limits_{i_0 \neq i} \left(s\left(i_0,\ i\right) \cdot b_{i,j}\right)}{\sum\limits_{i_0 \neq i} s\left(i_0,\ i\right)}$$

Algorithm 3: Minimisation par voisinage

Data: La matrice B et un entier k.
Result: La matrice approchée $\widehat{A}$.
$S \longleftarrow BB^{\mathrm{T}}$;
for $i \in [\![1,\, n]\!]$ **do**
  $n\,[i] \longleftarrow \|B\mathrm{e}_i\|_2$
for $i \in [\![1,\, n]\!]$ **do**
  for $k \in [\![1,\, n]\!]$ **do**
    $S\,[i,\, k] \longleftarrow S\,[i,\, k]\,/\,(n\,[i]\,.n\,[k])$;
$\widehat{A} \longleftarrow S.B$;
$somme \longleftarrow S.(1)$;
for $i \in [\![1,\, n]\!]$ **do**
  for $j \in [\![1,\, m]\!]$ **do**
    $A\,[i,\, j] \longleftarrow A\,[i,\, j]\,/somme\,[i]$;
return $\widehat{A}$;

4 Décomposition en valeurs singulières

Le but est ici de faire apparaître k modèles de personnes, dont toutes les autres seront des combinaisons linéaires.

Autrement dit, on souhaite faire apparaître la matrice $\widehat{A}$ de la manière suivante :

$$\widehat{A} = P \cdot O$$

avec $P \in \mathcal{M}_{n,\,k}(\mathbb{R})$ de colonnes libres que l'on peut prendre orthonormées et $O \in \mathcal{M}_{k,\,m}(\mathbb{R})$.

On peut facilement accéder à une telle décomposition à l'aide de la décomposition en valeurs singulières.

Algorithm 4: Recommandation par décomposition en valeurs singulières

Data: La matrice B et un entier k.
Result: La matrice approchée $\widehat{A}$.
$[U,\, S,\, V] \longleftarrow \mathrm{SVD}\,(B)$;
for $i \in [\![k,\, \mathrm{size}\,(S)]\!]$ **do**
  $S\,[i,\, i] \longleftarrow 0$;
$\widehat{A} \longleftarrow USV^{\mathrm{T}}$;
return $\widehat{A}$;