Rapport Projet annuel

Sujet : Comment sont vos préférences par rapport à celles de vos amis ?

De : AUBRY Nicolas, LEPETIT Lucie, SOROOSH MARKAZI Navid

Professeur : CREMILLEUX Bruno

Année 2021-2022
Master 2 Informatique DOP
Université de Caen Normandie

Sommaire:

Introduction :

De nos jours, un grand nombre de données sont étudiées dans les fouilles de données. Dans notre cas, nous étudions des sous-groupes dans un ensemble de données.
L'objectif de notre projet était d'étudier la recherche de sous-groupes spécifiques pour un comportement dit ”intéressant" selon le comportement d'une population.
Cela peut être utile pour des entreprises afin de cibler des comportements qui sortent de la moyenne. Elles vont pouvoir alors s'ajuster en prenant en compte leurs descripteurs (âge, ville …), le tout afin de répondre au mieux au besoin.
Pour notre étude, nous avons utilisé l'article scientifique, intitulé “Discovering a taste for the unusual : exceptional models for preference mining”, créé par Cláudio Rebelo de Sá, Wouter Duivesteijn, Paulo Azevedo, Alípio Mário Jorge, Carlos Soares et Arno Knobbe.

I/Contexte

A/ Les données

Les données que nous avons utilisé sont issues d'un ensemble de préférences pour des sushis, "SUSHI Preference Data Sets[1]".
Les données ont été constituées en donnant un questionnaire à 5000 individus.
Ici, les individus sondés étaient présents au Japon, ils sont soit issus de ce pays, soit de pays étrangers. Ils devaient ordonner 10 sushis (sur un total de 10 sushis ou bien sur 100 sushis) selon leur préférence. Les détails sur la constitution de ce questionnaire sont données en Annexe I/.

Les données sont constituées d'un ensemble de sushis, qui contiennent chacun un identifiant, de 0 à 100 avec leurs détails. Mais on retrouve aussi des fichiers contenant les informations/descripteurs pour chacune des personnes questionnées. (Détails en Annexe II/.)

Ce sont ces descripteurs que nous utiliserons par la suite pour la recherche de différents sous-groupes.

Mais pourquoi utiliser spécifiquement l'ensemble de données des sushis ? Nous l'avons sélectionné car c'est une base de données contenant des ranking de préférence, qui est l'objet de notre étude.

B/La recherche de sous groupe

- ***La méthode classique***

En fouille de données, un sous groupe est définie par une classe. Ci-après dans le Tableau 1 se trouve une population de différents individus.

Tableau 1 d'une population d'individus

Individu	ville	âge
P1	caen	22 moyen
P2	cheux	15 jeune
P3	cheux	65 vieux
P4	carpiquet	32 moyen
P5	caen	17 jeune
P6	caen	44 vieux

Par exemple, selon le Tableau 1, le sous groupe défini par la classe "habitant de Cheux" est composé des individus {P2,P3}.

[1] https://www.kamishima.net/sushi/

Le cas spécifique des préférences

Individu	ville	âge	préférences de Sushi
P1	caen	22 moyen	s1 > s3 > s2
P2	cheux	15 jeune	s2 > s3 > s1
P3	cheux	65 vieux	s3 > s2 > s1
P4	carpiquet	32 moyen	s1 > s2 > s3
P5	caen	17 jeune	s2 > s3 > s1
P6	caen	44 vieux	s3 > s2 > s1

Tableau 2 d'une population d'individus, avec un ranking de sushi

Dans notre étude, nous utilisons en plus des "ranking", (que vous pouvez trouver en dernière colonne dans le Tableau 2), qui sont un ordonnancement des préférences d'un certain objet par des individus.

Un **sous-groupe** ici est un ensemble d'individus ayant certains descripteurs en commun.

Un **descripteur** est un attribut, propre à un individu dans notre cas, comme sa ville, ou encore son âge ou son genre.

Ici, l'objet est le sushi présent sous plusieurs variantes, et les individus sont des répondants au questionnaire. Mais il existe aussi d'autres bases qui pourraient être étudiées. Par exemple, dans l'article qui nous a servi de base à notre étude, deux ensembles de données sont intéressants: "GermanElections2009" ainsi que le celui des "Top7Movies" car ils sont basés sur les mêmes principes, avec un classement de candidats pour les élections allemandes, et d'un classement de 7 films pour le second ensemble.

C/ EPM: Exceptional Preference Mining.

EPM sert à la recherche de sous groupes avec des préférences dites "déviantes".
Ce sont les mesures d'EPM qui nous permettront de réaliser l'algorithme sur cette recherche de sous-groupes spécifiques que nous devons implémenter sur les données de sushis.

Nous pouvons obtenir plus d'informations sur les préférences des sushis grâce à différentes méthodes de calculs.

Exemple avec un dataset de sushis, avec 10 individus :

Personnes	Attributs			Préférences
	Âge	Genre	Ville	
P1	22	H	Caen	S1 > S3 > S4 > S2
P2	49	F	Paris	S2 > S4 > S3 > S1
P3	18	H	Lyon	S3 > S2 > S1 > S4
P4	36	H	Rouen	S1 > S2 > S4 > S3
P5	17	F	Caen	S2 > S3 > S4 > S1
P6	29	F	Rennes	S4 > S1 > S3 > S2
P7	35	H	Paris	S1 > S4 > S3 > S2
P8	56	F	Rennes	S2 > S4 > S1 > S3
P9	27	F	Lyon	S3 > S4 > S2 > S1
P10	44	F	Caen	S4 > S3 > S2 > S1

Voici pour illustrer nos propos, un exemple de classement de préférences de sushis de 10 individus. On peut voir, par exemple, que l'individu P1 préfère, dans l'ordre d'abord le sushi numéro 1, puis le numéro 3 etc.

On a donc ici, pour tous ces individus, quatres sushis différents, ordonnés dans un ordre croissant, du plus préféré au moins préféré.

L'utilisation de matrices de préférence

Des matrices de préférences doivent être réalisées afin de nous permettre de calculer des sous-groupes qui seraient intéressants.

- Le calcul de la matrice de préférence pour un individu

Nous devons dans un premier temps, créer la matrice de préférence pour chacun des individus. Pour l'ensemble de sushis : {s1,s2,s3s,s4}, voici la matrice de préférence de l'individu P1 :

	s1	s2	s3	s4
s1	0	1	1	1
s2	-1	0	-1	-1
s3	-1	1	0	1
s4	-1	1	-1	0

Dans cette matrice, chaque ligne et colonne représentent un sushi.
Ici, nous croisons chacun des sushis par paire, en prenant le sushi en ligne, et en le croisant avec celui en colonne.

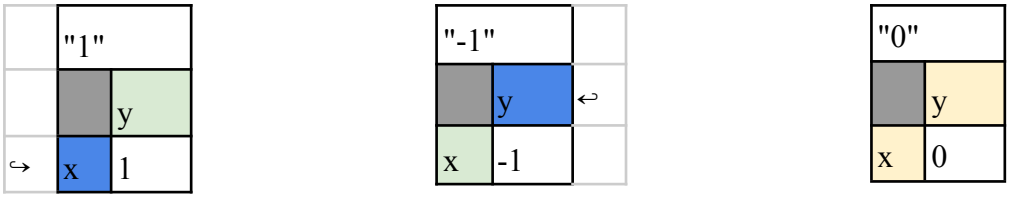


Figure 1 : Indication sur les résultats de préférence à poser

Au croisement de 2 sushis, nous plaçons un nombre spécifique, comme vous pouvez le voir sur la Figure 1 :

- 1 : Ce nombre signifie que le sushi en ligne est préféré au sushi en colonne, ce sushi vient avant l'autre dans l'ordre de préférence.

-1 : Au contraire ici, ce nombre suggère que le sushi en ligne est moins préféré à celui en colonne, donc ce n'est pas celui qui est préféré, il vient après dans l'ordre de préférence.

0: ce nombre est posé s'il n'y a pas d'information sur la préférence en comparant les 2 sushi en ligne et en colonne. Aussi, dans notre cas d'exemple, il représente la diagonale de la matrice, le sushi n°1 ne peut pas être comparé à lui même par exemple.
Avec cette matrice de préférence de l'individu P1, on peut voir qu'il préfère davantage le sushi n°1 par rapport aux autres.

- Le calcul de la matrice de préférence de la population :

Après avoir obtenu l'ensemble des matrices de préférence de chacun des individus, nous pouvons calculer la matrice de préférence moyenne pour la population "MD"(Figure 3).

Préférence	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Mean
s1 - s2 :	1	-1	-1	1	-1	1	1	-1	-1	-1	-0,2
s1 - s3 :	1	-1	-1	1	-1	1	1	1	-1	-1	0
s1 - s4 :	1	-1	1	1	-1	-1	1	-1	-1	-1	-0,2
s2 - s1:	-1	1	1	-1	1	-1	-1	1	1	1	0,2
s2 - s3 :	-1	1	-1	1	1	-1	-1	1	-1	-1	-0,2
s2 - s4 :	-1	1	1	1	1	-1	-1	1	-1	-1	0
s3 - s1 :	-1	1	1	-1	1	-1	-1	-1	1	1	0
s3 - s2 :	1	-1	1	-1	-1	1	1	-1	1	1	0,2
s3 - s4 :	1	-1	1	-1	1	-1	-1	-1	1	-1	-0,2
s4 - s1 :	-1	1	-1	-1	1	1	-1	1	1	1	0,2
s4 - s2 :	1	-1	-1	-1	-1	1	1	-1	1	1	0
s4 - s3 :	-1	1	-1	1	-1	1	1	1	-1	1	0,2

Figure 2 : Calcul des moyennes pour chaque individu et sushi.

Ici, dans la Figure 2, on calcule chacun des résultats selon toute la population, qui est l'ensemble des individus de nos données.

Figure 3 : Matrice MD

	s1	s2	s3	s4
s1	0	-0.2	0	-0.2
s2	0.2	0	-0.2	0
s3	0	0.2	0	-0.2
s4	0.2	0	0.2	0

On fait la moyenne de la somme des nombres de préférence calculés, et on reporte chaque ligne de moyenne sur la matrice. Voici son résultat à gauche.

Dans cette matrice moyenne MD, on peut dire que c'est le quatrième sushi qui est le plus préféré parmi l'ensemble des individus, car il n'y a aucun nombre négatif sur la quatrième ligne de la matrice.

Calcul de la déviation selon un sous-groupe

Après avoir obtenu la matrice de préférence de chacuns des individus, et la matrice de préférence moyenne de la population “MD”, nous pouvons calculer sa déviation.

Dans un premier temps, nous avons besoin d’un sous-groupe, nous définissons un seuil afin qu’il y ait un minimum d’individus présentés.

Ici, pour l’exemple, nous avons choisi de prendre les individus qui vivent à Caen. Il y a 3 individus : P1,P5,P10.

Avec la méthode précédente qui calcule la moyenne des préférences, on peut générer la matrice moyenne pour ce sous-groupe.

P1, P5, P10 : subgroup from Caen				
Preference	P1	P5	P10	Mean
s1 - s2 :	1	-1	-1	-0,33
s1 - s3 :	1	-1	-1	-0,33
s1 - s4 :	1	-1	-1	-0,33
s2 - s1:	-1	1	1	0,33
s2 - s3 :	-1	1	-1	-0,33
s2 - s4 :	-1	1	-1	-0,33
s3 - s1 :	-1	1	1	0,33
s3 - s2 :	1	-1	1	0,33
s3 - s4 :	1	1	-1	0,33
s4 - s1 :	-1	1	1	0,33
s4 - s2 :	1	-1	1	0,33
s4 - s3 :	-1	-1	1	-0,33

Voici le résultat de cette matrice, on l’appelle “Ms” :

	s1	s2	s3	s4
s1	0	-0,33	-0,33	-0,33
s2	0,33	0	0,33	-0,33
s3	0,33	0,33	0	0.33
s4	0.33	0.33	-0.33	0

Pour ce sous-groupe spécifique, on peut voir que c’est le sushi n°1 qui est le moins préféré, dû aux nombres négatifs sur la première ligne de Ms.

La déviation :

La déviation est un score permettant de savoir si le sous-groupe est ou non intéressant. Pour ce faire, nous avons besoin de la matrice de distance LS et d’une mesure de qualité.

La matrice de distance “LS” représente le calcul de la matrice de distance, entre la matrice moyenne de la population “MD” et la matrice moyenne du sous-groupe spécifique “Ms” sélectionné. **LS = ½ * (MD - Ms)**

Les mesures de qualité

Après avoir obtenu la matrice de distance, on peut calculer la déviation grâce à des mesures de qualité. Il y a trois différentes mesures :

1) RWNorm: “Rankingwise Norm quality measure” :

$$\mathrm{RWNorm}(S) = \sqrt{s/n} \cdot ||L_S||_F = \sqrt{s/n} \cdot \sqrt{\sum_{i=1}^{k} \sum_{j=1}^{k} L_S(i,j)^2}$$

Cette mesure utilise la norme de Frobenius de LS en tant que mesure de distance. C’est la racine carrée de la somme sur tous les i,j de (A i,j)^2 au carré

2) LWNorm: “Labelwise Norm measure” :

$$\mathrm{LWNorm}(S) = \sqrt{s/n} \cdot \max_{i=1,\ldots,k} \sqrt{\sum_{j=1}^{k} L_S(i,j)^2}$$

3) Pairwise Measure :

$$\mathrm{PWMax}(S) = \sqrt{s/n} \cdot \max_{i,j=1,\ldots,k} |L_S(i,j)|$$

Vous pourrez trouver un exemple complet de calcul de ces mesures en Annexe III.
Avec ces trois mesures, si le score est haut, c’est qu’il dévie par rapport aux préférences de l'ensemble de la population. Plus il est proche de 0, plus le sous-groupe à le même comportement que la population d’origine.

II/ Contributions et Expérimentations

Pour l'utilisation de l'ensemble de données des sushis, nous avons dû dans un premier temps étudier les différentes données et fichiers. Ensuite nous devions les extraire afin de les utiliser pour créer un algorithme de recherche de sous groupes intéressants.
Enfin, après la réalisation de l'extraction et de l'algorithme, nous avons pu faire différentes expérimentations.

A/Prétraitement des données

Ayant besoin pour notre sujet du ranking des différents sushis, pour récupérer les données, nous avons utilisé les 2 fichiers de ranking de sushis, ainsi que les fichiers de descripteurs de sushis et d'utilisateurs. Par la suite, nous utilisons uniquement l'ensemble de données "itemset A".

Réalisation d'un "parser"

Pour notre parser générique, on trouvera trois classes principales : Item, User, Parser

Parser :
Le parser est une classe abstraite, "Parser", il permet de récupérer et de parser les données d'un (de) fichier(s) de manière générique.
Ce parser permet de remplir les données d'objets "Item", et les données d'utilisateur "User".

Person :
Person est une classe générique, on a ici les informations d'une personne étudiée.

Item :
Item est une classe générique, on a ici les informations d'un objet étudié.

Spécificité aux données de sushi

Pour notre ensemble de données de sushi, un parser, " ParserSushiSet" , sous classe de Parser a été créé, afin d'obtenir les données et descripteurs spécifiques aux données des sushi.

Une sous classe spécifique de Person, "User ", ainsi qu'une sous classe spécifique d'Item, " Sushi", sont présentes, et spécifiques à nos données.

En récupérant chacune des données, on remplit une liste de sushis avec leurs informations, et une liste d'utilisateurs, les 5000 individus du questionnaire, qui contiennent chacun une liste de sushis ordonnés selon leurs préférences, du moins aimé au plus préféré.
Notre parser récupère aussi l'ensemble des descripteurs possibles présents sur chaque individu.

De plus, le parser permet de créer des matrices de préférence pour chacun des individus de notre ensemble de données.

Par la suite, grâce à ces matrices de préférence, générées en instances de type de classe "Matrix", que nous détaillerons plus bas, nous pouvons réaliser les différents calculs et la méthode pour l'algorithme pour la recherche de sous-groupes intéressants.

B/ Algorithme de recherche de sous-groupe intéressant : BeamSearch pour EPM

Nous nous sommes servis de l'algorithme EMM "Exceptional Model Mining", présent dans l'article étudié, pour créer celui pour EPM, que nous vous présentons ci-après sur l'Algorithme 1.

Algorithme 1- Beam Search pour EPM :

Algorithm 1 BeamSearch for Exceptional Preference Mining

Input DatasetPopulation P, DatasetExistingDescriptors D, QualityMeasure q, MinimumFrequency f, BeamWidth x, BeamDepth y, SetDescriptors setD

Output resultGroupSet

```
1:  procedure SEARCHK
2:      currentDepth ← 0;
3:      candidateQueue ← newPriorityQueue();
4:      candidateQueue.add(populationasrootGroup);
5:      while (currentDepth ¡ y) do
6:          Integerkey ← currentDepth;
7:          PriorityQueuecandidateQueueTmp ← newPriorityQueue;
8:          for all descriptorValue in D from key  do
9:              PriorityQueue candidateQueueCP ← newPriorityQueue;
10:             candidateQueueCP.enQueue(candidateQueue);
11:             Mapselection ← newMap();
12:             selection.add(key, descriptorValue);
13:             while (candidateQueueCP != 0) do
14:                 Groupgroup ← candidateQueueCP.dequeue();
15:                 MapdescOfGroup ← newMap();
16:                 descOfGroup.add(key, descriptorValue);
17:                 Groupsubgroup ← group.generateSubGroup(f, selection);
18:
19:                 if subgroup! = null then return
20:                     calculateScoreofsubgroupwithq
21:                     candidateQueueTmp.add(sousgroupe);
22:                 end if
23:             end while
24:             end for
25:             PriorityQueuecandidateQueue2Step ← newPriorityQueue();
26:
27: add candidates from previous step
28:             CandidateQueue2Step.add(all candidateQueue);
29:
30: add all candidates from current step, but only x candidates to get only that beamWidth
31:             candidatesQueue2Step.add(all x first from candidateQueueTmp);
32:
33: add best candidates from current and previous step;
34:             candidatesQueue.add(candidateQueue2Step.dequeue());
35:             currentDepth ← currentDepth + 1;
36:         end while
37:         return set of best groups from candidateQueue;
38:     end procedure
```

Pour réaliser cet algorithme de recherche en faisceau, nous avons besoin comme pour EMM, d'une mesure de qualité et d'un ensemble de données.

Pour nous, cet ensemble de données est la liste de la population entière présente dans le "dataset" des sushis, où on y retrouvera les différentes matrices de "ranking"/préférence. Il nécessite aussi en

paramètres une profondeur et une largeur de faisceau, si l'on souhaite limiter la recherche à un nombre spécifique de sous branches.

Classe "BeamSearch" :
Cette classe permet d'instancier l'algorithme, avec en paramètres :

- Les données récupérées lors de l'étape du parser :
    1) La liste de personnes de la population de l'ensemble de données
    2) L'ensemble des descripteurs possibles existant dans l'ensemble de données
- Avec un seuil minimum d'individus pour les sous groupes,
- Et avec le type de méthode pour calculer le score de déviation.
    1) "1" pour la méthode de labelWise
    2) "2" pour la méthode de pairWise
    3) "3" pour la méthode de rankingWise

Nous avons deux variantes pour cet algorithme:

- la méthode "*searchPTimer()*" qui arrête la recherche à partir d'un temps donné en paramètre, et qui parcourt l'ensemble de la liste des descripteurs existants , selon une profondeur et une largeur en paramètres. Cette méthode renvoie une liste des dix meilleurs sous-groupes actuellement possibles lors de l'arrêt du temps.
- la méthode "*searchK()*" où il faut définir la liste de descripteurs que l'on souhaite prendre au départ en paramètres, et aussi avec une largeur et une profondeur souhaitée. Cette méthode renvoie une liste des meilleurs sous-groupes lorsqu'elle a fini d'être exécutée.

Trois sous-types de méthodes ont été réalisés avec la seconde variante "*searchK()*" :
Nous avons au départ, pour notre ensemble de données de sushi, une liste de dix descripteurs qui vont de 0 à 9. Chacun de ces trois sous-types prend en paramètre le nombre k, qui est le nombre de descripteurs souhaités à sélectionner.

- Le premier sous-type de méthode, qui renvoie une liste intitulée "kFirst". Il est celui qui réalise la recherche en fonction d'un nombre k , et donne les k premiers descripteurs selon la liste originale, grâce à la méthode "*getKFirstDescriptors()*". Cette liste est ensuite envoyée pour la méthode "*searchK()*".
- Le second sous-type de méthode, qui renvoie une liste intitulée "kRandom". Il prend un nombre k de descripteurs de manière aléatoire selon la liste originale et donne une liste de k descripteurs, grâce à la méthode "*getKRandomDescriptors()*". Cette liste est ensuite envoyée pour la méthode "*searchK()*".
- Le troisième type, qui renvoie une liste intitulée "kBest", prend d'abord les meilleurs descripteurs selon la liste, en fonction du premier sous niveau, avec la méthode "*getKBestDescriptors()*".
  "*getKBestDescriptors()*" utilise la méthode "*searchPBest*()" qui prend uniquement un descripteur en paramètre et fait de même pour l'ensemble des k descripteurs de la liste originale, il s'exécute à la profondeur numéro une, et ensuite, il fait la méthode de recherche "*searchK()*" selon les k meilleurs descripteurs de départ trouvés .

D'autre part, cet algorithme utilise la classe "Group" que nous avons réalisée, et il retourne une liste de groupes, dix groupes nous ont été demandés par notre professeur.

Classe “Group” :

Cette classe permet d’obtenir différentes données pour des sous-groupes.
D’une part, avec la méthode *“generateSubGroup()”*, grâce à des descripteurs spécifiques demandés présents dans une “Map”, par exemple {0=1, 3=2} et à une liste d’individus, et à un seuil minimum d’individus, on peut créer un sous-groupe d’individus qui possèdent ces deux descripteurs choisis.
On peut aussi obtenir la taille de la population du groupe, mais aussi calculer son score.

Le score est réalisé à partir de deux classes :

Classe “Matrix” :

Cette classe permet de générer une matrice de ranking, et de réaliser différents calculs de matrices.

Classe “Deviation” :

Cette classe permet de calculer la déviation, donc le score d’un sous-groupe, à l’aide des trois mesures de qualité que nous vous avons présentées. Différents calculs sont faits, avec des matrices de nombre de type “Double” utilisés par la classe “Matrix”.

Fonctionnement de l’algorithme :

Notre algorithme de BeamSearch pour EPM récupère dans un premier temps une liste de descripteurs pour savoir quels descripteurs seront à utiliser.
Ensuite il utilise une PriorityQueue pour récupérer les meilleurs sous groupes, avec un score décroissant.
Pour le premier niveau de profondeur, on récupère les scores des différents sous-groupes possibles selon le sous-groupe de population d’origine.

Il passe ensuite aux autres sous-niveaux selon la profondeur demandée, et utilise une autre PriorityQueue, qui trie et met au fur et à mesure les sous-groupes des sous-branches, pour à la fin, soumettre les meilleurs groupes.
On récupère à la fin les dix meilleurs sous-groupes selon toutes les profondeurs parcourues.

Voici ci-après sur le Schéma 1 le déroulement de l’algorithme de recherche en faisceau pour EPM :

Schéma 1 : Déroulement de l'algorithme Beam Search

Note : Sur l'encadré en pointillés rouge, on prend ici les 3 meilleurs scores car la largeur du faisceau est de 3, on analyse ensuite les sous-groupes provenant de ces branches uniquement, (entourés en vert), et ce pour chaque profondeur à analyser.

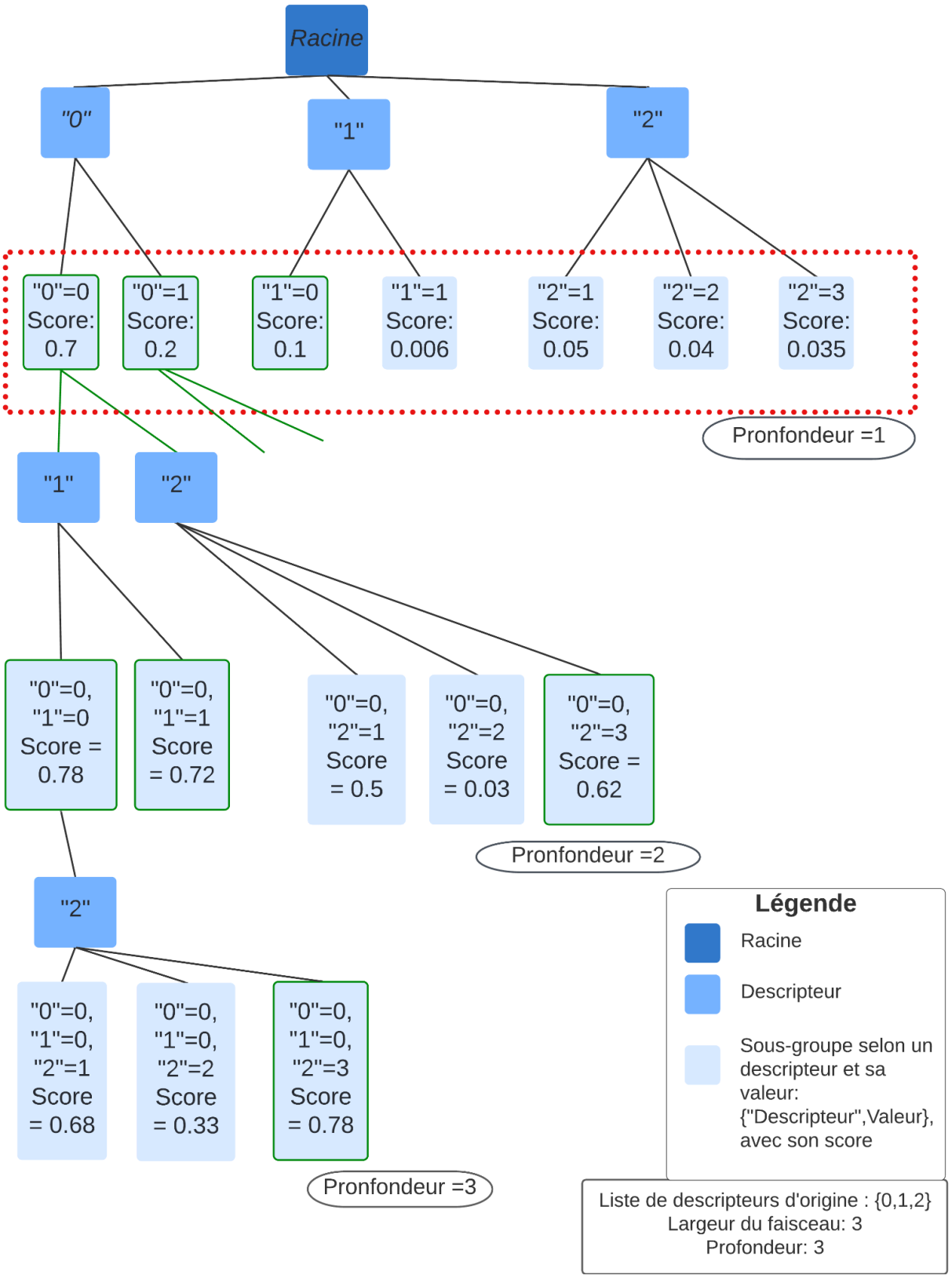
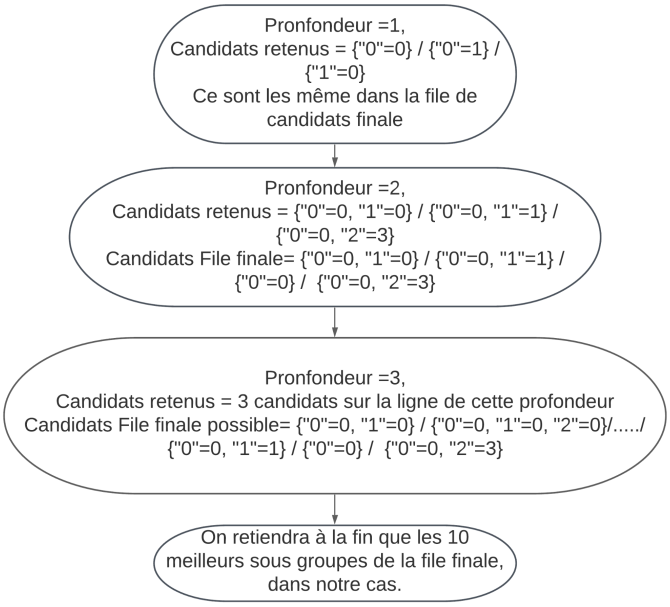


Schéma 2 : Détail de la liste de candidats pris pour chacune des profondeurs.



On peut voir dans le Schéma 2, qu'il est possible que dans la liste finale de sous-groupes, il reste des sous-groupes des niveaux supérieurs.

C/ Expérimentations des résultats

Pour l'exécution de notre algorithme de recherche en faisceau pour EPM, voici les différents résultats obtenus.

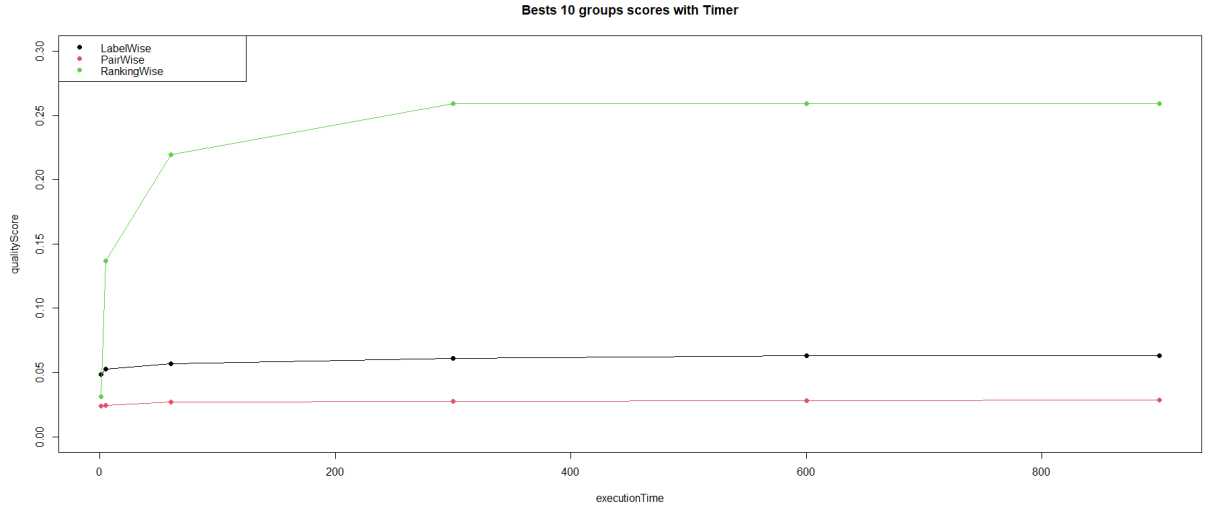
Nous avons récupéré les résultats obtenus par les trois différentes mesures de qualité.
Pour chacuns de nos résultats, nous avons en paramètre fixé la fréquence minimum de personnes, qui est de 10 personnes, la liste totale d'individus de la population est de 5000 personnes.

Exécution à l'aide d'un stoppeur de temps:

Aussi nous avons pris les résultats du BeamSearch, qui possède une profondeur et une largeur infinies, donc qui fait en quelque sorte un parcours en largeur, et nous l'avons stoppé à plusieurs différents temps en secondes. Mais nous avons regénéré la recherche en faisceau pour chaque temps, afin de voir si l'algorithme pouvait fonctionner différemment lors d'une nouvelle exécution même si elle est plus longue, et cela n'a pas été le cas, les résultats restent bien les mêmes.

Voici un aperçu des résultats avec la méthode du stoppeur de temps :

Courbe 1 : Moyenne des scores des 10 meilleurs groupes selon le temps



Nous avons ici calculé le score moyen, que nous nommons le quality Score, en fonction des dix sous-groupes trouvés. La courbe suivante présente sur l'axe des X le quality Score, et en axe des ordonnées le temps d'exécution de la méthode de recherche en secondes.

On peut nettement voir que la méthode de RankingWise norme trouve des sous-groupes de plus en plus intéressants lorsque qu'il parcourt de plus en plus les différentes branches en largeur puis en profondeur, avec la courbe verte, et on constate qu'au bout de 300 secondes, il ne trouve plus de nouveaux sous-groupes avec de haut scores, donc intéressants, car la courbe est stagnante.

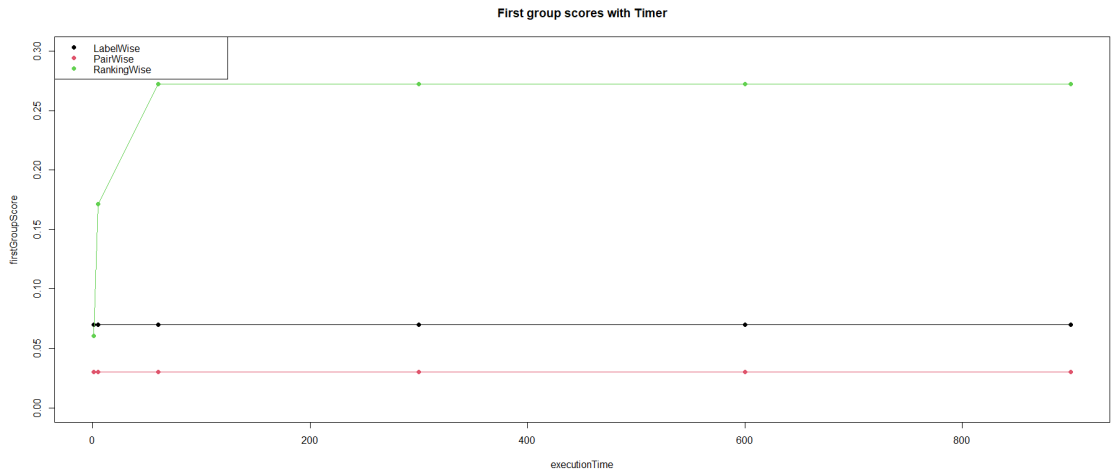
Pour les méthodes de LabelWise et PairWise, on pourrait se demander si elles sont moins efficaces pour trouver des sous groupes intéressants, car les deux courbes noire et rouge commencent à avoir un quality Score moyen qui stagne dès le début. Soit ils trouvent directement un sous-groupe qui est intéressant, ou bien ils ne permettent pas d'en trouver de meilleurs.

À l'écriture de ce rapport, nous nous demandons si l'implantation de ces deux méthodes ont bien été réalisées, car sur notre exemple concret présent en Annexe III, nous retrouvions principalement les mêmes sous-groupes intéressants avec les trois méthodes. Mais cela peut aussi être dû au fait que notre exemple était réalisé sur un ensemble très petit de 10 individus contre 5000 pour le dataset des sushi.

Nous vous présentons la Courbe 2 ci-après, avec la même méthode de stoppeur de temps, mais cette fois-ci en ne récupérant que le meilleur groupe trouvé et son score.

On peut voir que les courbes s'alignent de la même façon que les courbes précédentes. Donc les sous groupes trouvés ont des scores très proches du score du premier sous groupe.

Courbe 2 : Scores des premiers meilleurs groupes selon le temps



Exécution à l'aide des trois sous types de méthodes et des profondeurs :

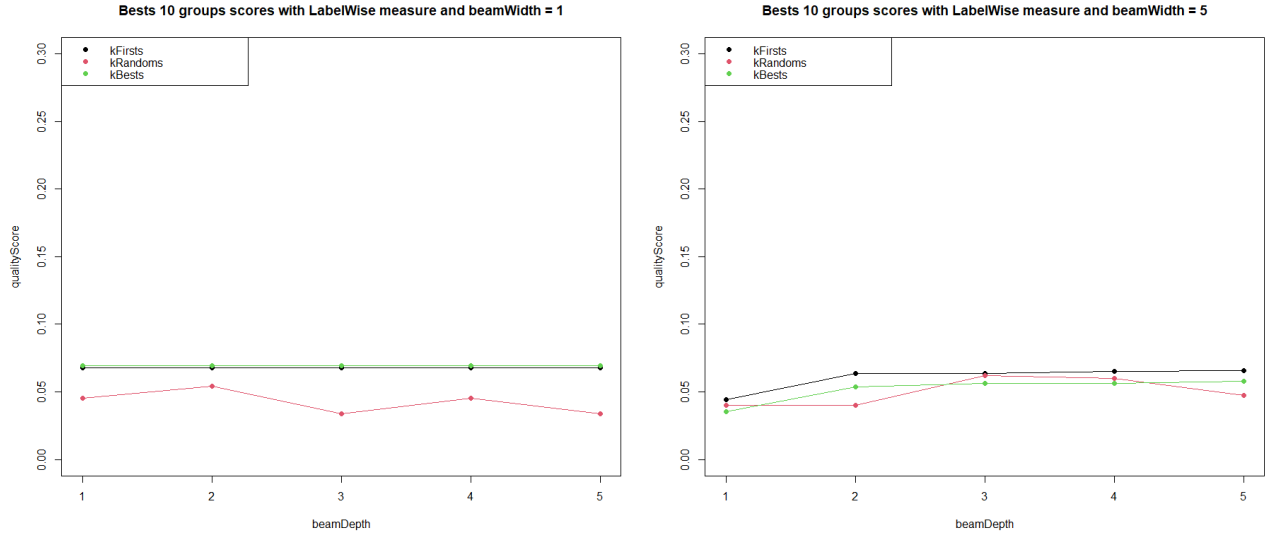
Maintenant, nous cherchons à voir s'il y a une différence avec une même largeur fixée, mais avec une profondeur plus grande, selon les trois sous-types de méthodes explicitées dans la partie II/B à la page 13.

Nous avons exécuté les trois sous-types de méthodes, avec une profondeur fixe de 1 et de 5, et les avons analysées en fonction de la profondeur qui augmente de 1 à 5.

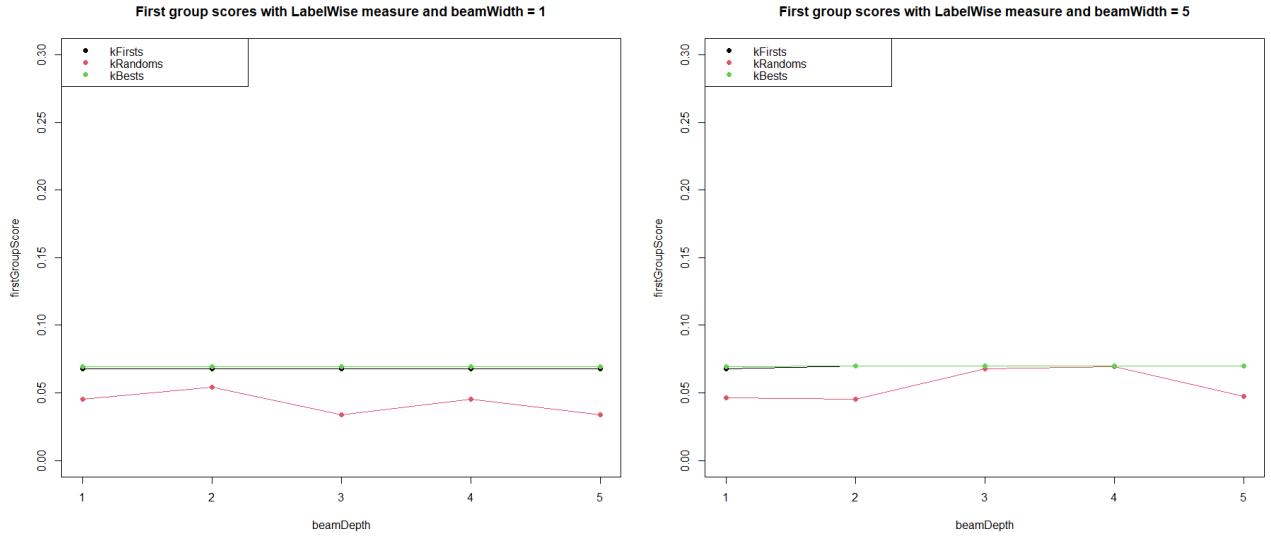
La méthode KFirst, est en noir, la méthode kRandom en rouge, et la méthode kBests en vert.

Voici les résultats que nous obtenons avec le score moyen des 10 sous-groupes trouvés et le score du meilleur sous-groupe :

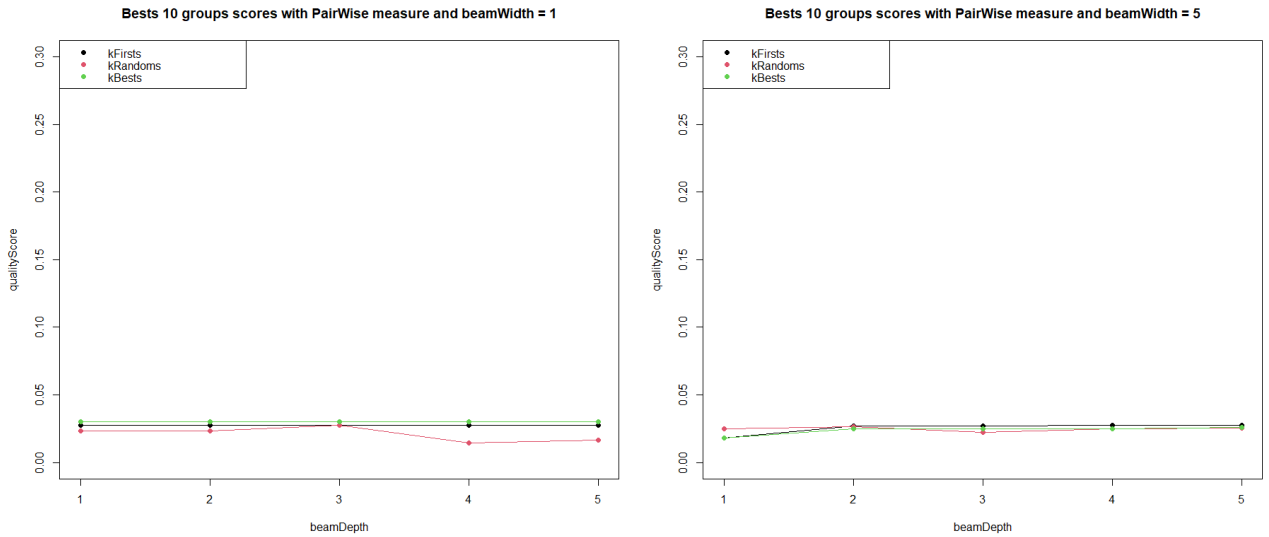
Courbe 3 et 4 : Moyenne des scores des 10 meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de LabelWise :



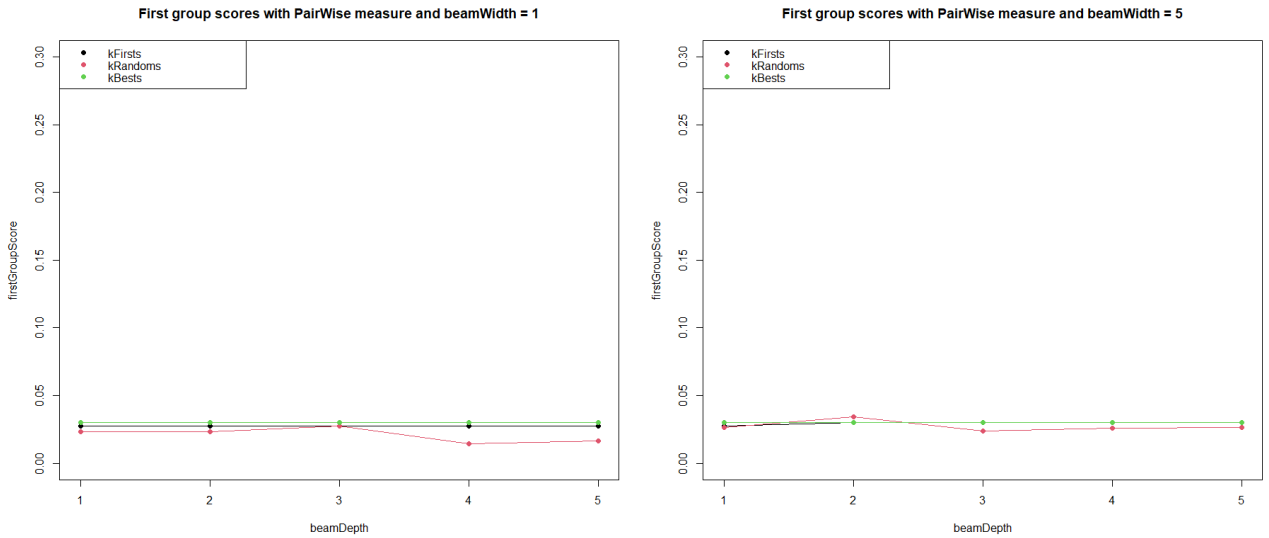
Courbe 5 et 6 : Scores des premiers meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de LabelWise :



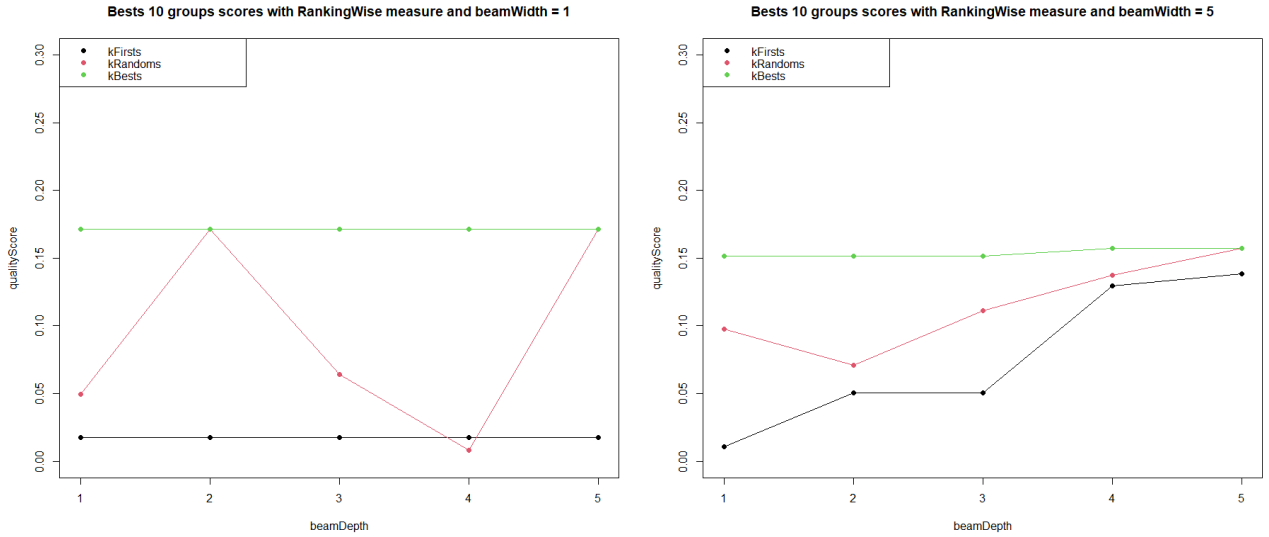
Courbe 7 et 8 : Moyenne des scores des 10 meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de PairWise



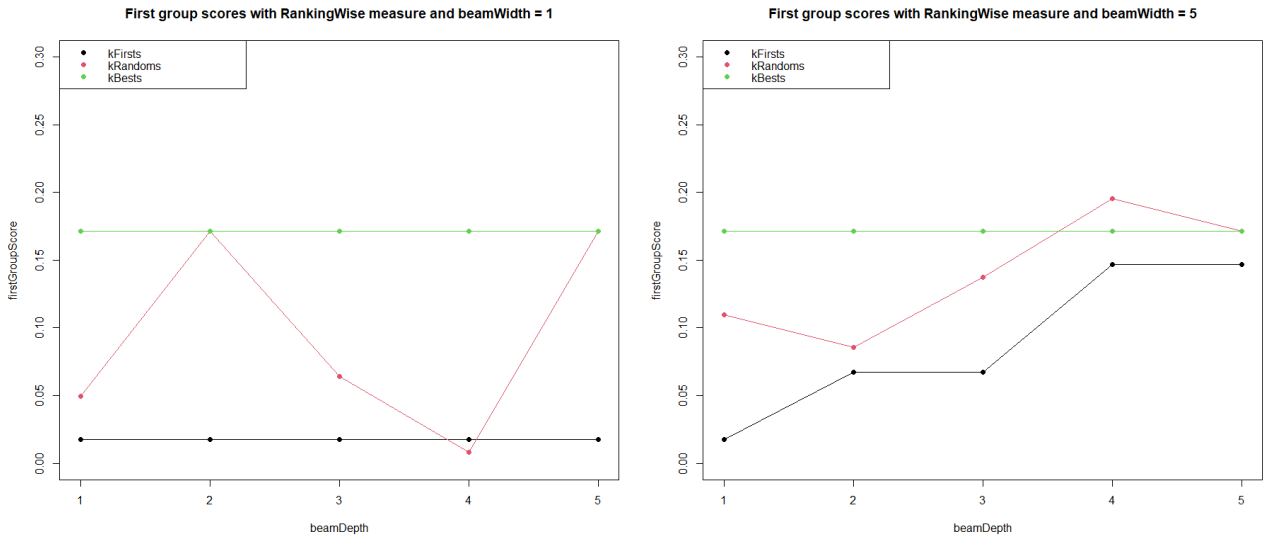
Courbe 9 et 10 : Scores des premiers meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de PairWise :



Courbe 11 et 12 : Moyenne des scores des 10 meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de RankingWise



Courbe 13 et 14 : Scores des premiers meilleurs groupes selon la profondeur, avec les 3 sous-types de méthode, avec la méthode de RankingWise :



En comparant les résultats des différentes courbes de 10 meilleurs sous-groupes et celui du premier sous-groupe, on peut affirmer l'hypothèse que les meilleurs sous-groupes générés ont bien un score proche du premier, il ne faut pas tenir compte des courbes fluctuantes générées avec la sous-méthode de “kRandom” car il peut y avoir des descripteurs sélectionnés qui sont très bons ou bien très mauvais pour la recherche de faisceau.

Sur les courbes 3 et 4, on peut voir encore une fois que la méthode de LabelWise stagne du début, avec la profondeur de 1 jusqu'à la fin avec la profondeur de 5, aussi bien avec une largeur de 1 ou de 5, avec n’importe lequel des trois sous-types de méthodes. On peut noter les mêmes remarques pour la méthode de PairWise. Bien qu’à la courbe 4, on voit une hausse de score donc de meilleurs sous-groupes ont été trouvés avec une profondeur de 2 et une largeur de 5 pour les sous-méthodes “kFirst” et “Kbest”.

En revanche pour la méthode de RankingWise sur les courbes 12 et 14, on constate que pour les méthodes “kFirst” et “kRandom”, plus la profondeur augmente plus le score est haut. On peut aussi dire que la méthode “kBest”, qui prend les meilleurs descripteurs pour la profondeur 1 n’est pas le meilleur moyen d’obtenir des sous-groupes intéressants, on peut en conclure ici que les sous-groupes intéressants sont majoritairement présents dans les plus hauts sous-groupes d’un arbre. Il peut par exemple y avoir beaucoup trop de personnes avec les mêmes préférences à ces niveaux.

On peut noter que le meilleur sous-groupe trouvé est un sous-groupe avec les descripteurs suivants : {1=4, 3=11}, son score est d’environ 0.27 ce qui est plutôt haut. Selon l’Annexe II, le descripteur 1 correspond à l'âge, la valeur de 4 correspond à un âge de 50 à 59 ans. Le descripteur 3 correspond à l’id de la préfecture où la personne a le plus vécu jusqu’à ses 15 ans. Ici, la préfecture 11 correspond à la préfecture de Saitama au Japon. Ce sous-groupe est composé de 10 individus, et il est bien intéressant selon nos termes.

III/Conclusion :

A/ Les problèmes rencontrés

Un des principaux problèmes que nous avons pu rencontrer, a été le manque de temps. En effet, nous avons eu du mal à comprendre le sujet dès le début. Nous ne connaissions pas bien certains éléments comme BeamSearch par exemple.

B/ La répartition du travail

Le travail a surtout été fait en groupe. Nous avons dans un premier temps, étudié les données, Lucie a réalisé le parser. Nicolas s'est chargé de réaliser les outils de calcul de Matrice et de déviation, qui ont plus au fur et à mesure été adaptés pour fonctionner avec les données. Ensuite Nicolas et Navid ont étudié la façon dont nous pouvions créer l'algorithme de BeamSearch, et nous avons tous implémenté la façon de le faire, en se concertant car nous avions au départ du mal à procéder à sa conception. Lucie a implémenté les différentes variantes de BeamSearch afin de pouvoir obtenir des expérimentations. Enfin nous avons réalisé ces expérimentations. Navid a transféré les résultats des expérimentations de manière automatisée dans un fichier de type classeur excel .csv, pour ensuite les utiliser dans le logiciel R afin de générer les graphiques que nous souhaitions.

C/ L'utilisation future:

La recherche de sous-groupes intéressants pourrait être remplacée par d'autres algorithmes. Ce projet pourrait être complété pour analyser en plus de l'analyse des utilisateurs et de leurs descripteurs ici, par l'analyse des informations sur les sushis, pour savoir si tel ou tel ingrédients présent à l'intérieur change ou non le ranking des sushis.
D'autre part, on pourrait voir son action sur d'autres ensemble de données, possiblement plus importants, pour analyser la performance de notre algorithme, et voir des sous-groupes intéressants qui dévient de manière importante.
L'intérêt subjectif, qui est la recherche d'un ensemble d'individus intéressant, pourrait aussi être une alternative. Le programme pourrait être utilisé par Monsieur Crémilleux, pour ses propres recherches car il n'existait pas pour lui d'outil pour l'Exceptional Preference Mining réel et qui fonctionne complètement.

Annexe

I/Questionnaire pour l'ensemble des données de sushi

Six étapes étaient nécessaires sur ce questionnaire :

1) La méthode de ranking n°1: "ranking method 1":
   L' individu devait trier les sushi de l'ensemble d'items A, selon ses préférences, dans un ordre croissant, donc du moins préféré au plus préféré ?
2) La méthode de notation: "scoring method":
   L' individu devait noter leur préférence sur les sushi de l'ensemble d'items B, sur une échelle de 5 points, donc de 1 à 5 ? dans les données ca va de -1 à 4
3) L' individu devait dire comment ils pensaient que le sushi était huileux
4) L' individu devait dire à quelle fréquence ils mangeaient ces sushi
5) La méthode de ranking n°2: "ranking method 2":
   L' individus devaient trier les sushis de l'ensemble d'items B, selon ses préférences, dans un ordre croissant, donc du moins préféré au plus préféré ?
6) L'individu devait répondre sur des informations géographiques.

II/ Composition des données et Étude des fichiers

-Les différents sushis et leurs identifiants

(à noter : les sushis de l'ensemble A sont présents dans l'ensemble B, mais certains n'ont pas les mêmes ID)

- Ensemble d'items A:

```
0:ebi  (shrimp)
1:anago (sea eel)
2:maguro (tuna)
3:ika (squid)
4:uni ( (sea urchin)
5:ikura (salmon roe)
6:tamago (egg)
7:toro (fatty tuna)
8:tekka_maki (tuna roll)
9:kappa_maki (cucumber roll)
(10 sushi en tout)
```

- Ensemble d'items B:

```
 0:ebi (shrimp)
 1:anago (sea eel)
 2:maguro (tuna)
 3:ika (squid)
 4:uni (sea urchin)
 5:tako (octopus)
 6:ikura (salmon roe)
 7:tamago (egg)
 8:toro (fatty tuna)
 9:amaebi (AMA shrimp)
10:hotategai (scallop)
ect.. (100 sushis en tout)
```

- fichiers d'information/descripteurs des données
  a) sushi3.idata:

Ceci représente l'ensemble des "features"/descripteurs pour chacun des 100 sushis de l'ensemble B. Une ligne représente un sushi avec comme descripteurs :

- 1) id
- 2) nom
- 3) style (0=maki, 1=autre)
- 4) groupe majeur (0=seafood, 1=autre), 0 = group mineur 0 à 8
- 5) groupe mineur 0 à 11

Les groupes mineurs sont les suivants :

- 0:aomono (blue-skinned fish)
- 1:akami (red meat fish)
- 2:shiromi (white-meat fish)
- 3:tare (something like baste; for eel or sea eel)
- 4:clam or shell
- 5:squid or octopus
- 6:shrimp or crab
- 7:roe
- 8:other seafood
- 9:egg
- 10:meat other than fish
- 11:vegetables

Les types de “vegetables” sont les suivants :

1) le poid de l'huile en goût (de 0(=huileux) à 4)
2) la fréquence que l’individu mange le sushi (de 0 à 3(=le plus fréquent))
3) le prix normalisé
4) la fréquence que le sushi est vendu dans le sushi shop, (de 0 à 1(=le plus fréquent))

b) sushi3.udata :

Ceci représente l’ensemble des "features"/descripteurs pour chacun des 5000 individus.
Chaque ligne est un individu. Chaque colonne est un descripteur, ce sont ces descripteurs que nous utilisons.

1) ID
2) genre 0:homme 1:femme
3) âge:
    - 0:15-19
    - 1:20-29
    - 2:30-39
    - 3:40-49
    - 4:50-59
    - 5:60-
4) temps de remplissage du questionnaire
5) préfecture ID , le plus vécut jusqu’à 15 ans
6) region ID , le plus vécut jusqu’à 15 ans
7) east/west ID , le plus vécut jusqu’à 15 ans
8) préfecture ID, vit actuellement
9) regional ID, vit actuellement
10) east/west ID, , vit actuellement
11) 0 si 5=8 donc vivent toujours au même endroit; 1 sinon

Pour notre travail, les données de descripteurs sont notées de 0 à 10, nous ne prenons pas en compte l’identifiant de la personne. Par exemple, 0 représente le genre d’une personne, il peut être écrit de la manière suivante: “0=1” si l’on veut choisir les personnes qui ont pour genre d’être une femme.

Voici les différents identifiants de préfecture :

0:Hokkaido	1:Aomori	2:Iwate
3:Akita	4:Miyagi	5:Yamagata
6:Fukushima	7:Niigata	8:Ibaraki
9:Tochigi	10:Gunma	11:Saitama
12:Chiba	13:Tokyo	14:Kanagawa
15:Yamanashi	16:Shizuoka	17:Nagano
18:Aichi	19:Gifu	20:Toyama
21:Ishikawa	22:Fukui	23:Shiga
24:Mie	25:Kyoto	26:Osaka
27:Nara	28:Wakayama	29:Hyogo
30:Okayama	31:Hiroshima	32:Tottori
33:Shimane	34:Yamaguchi	35:Ehime
36:Kagawa	37:Tokushima	38:Kochi
39:Fukuoka	40:Nagasaki	41:Saga
42:Kumamoto	43:Kagoshima	44:Miyazaki
45:Oita	46:Okinawa	47:foreign countries

Voici les différents identifiants de région et leurs préfectures liées :

```
 0:Hokkaido               { 0 }
 1:Tohoku                 { 1, 2, 3, 4, 5, 6 }
 2:Hokuriku               { 7, 20, 21, 22 }
 3:Kanto+Shizuoka         { 8, 9, 10, 11, 12, 13, 14, 16 }
 4:Nagano+Yamanashi  { 15, 17 }
 5:Chukyo                  { 18, 19, 24 }
 6:Kinki                   { 23, 25, 26, 27, 28, 29 }
 7:Chugoku                 { 30, 31, 32, 33, 34 }
 8:Shikoku                { 35, 36, 37, 38 }
 9:Kyushu                 { 39, 40, 41, 42, 43, 44, 45 }
10:Okinawa                 { 46 }
11:Forign                 { 47 }
```

Voici le type d'identifiant selon que la personne se trouve à l'est ou à l'ouest du Japon :

```
0 :Eastern Japan (region no is <= 5)
1 :Western Japan (region no>= 6)
```

- fichiers de ranking

a) sushi3a.5000.10.order:

Ceci est un fichier de ranking, contenant l'ordre croissant de préférence des sushi de l'ensemble A, obtenu avec la méthode de ranking n°1 (étape 1).

b) sushi3b.5000.10.order:

Ceci est un fichier de ranking, contenant l'ordre croissant de préférence des sushi de l'ensemble B, obtenu avec la méthode de ranking n°2 (étape 5)

Dans ces deux fichiers on trouve :

Une première ligne, qui contient le nombre de sushis présents.
Le reste des lignes est composé de l'ordre de préférence répondu.
Chacune des 5000 lignes correspond à un des 5000 individus.
Exemple : La ligne 5 de ce fichier = la ligne correspondante dans le fichier sushi3.udata, le fichier des descriptions des individus.

- fichier de notation

sushi3b.5000.10.score:

Ce fichier contient les scores de préférence des sushi de l'ensemble B, obtenu avec la méthode de notation (étape 2), il n'est pas utilisé pour nos travaux, car nous ne notons pas de la même manière.
Les différents scores sont ici : -1 = non noté; 0 = le moins aimé; 4 = le plus préféré.

III/ Cas concret complet avec utilisation de ces méthodes

Illustration complète sur un cas “simple” avec 3 sushi et 6 individus.

Individu	ville	age	préférences de Sushi
P1	caen	22 moyen	s1 > s3 > s2
P2	cheux	15 jeune	s2 > s3 > s1
P3	cheux	65 vieux	s3 > s2 > s1
P4	carpiquet	32 moyen	s1 > s2 > s3
P5	caen	17 jeune	s2 > s3 > s1
P6	caen	44 vieux	s3 > s2 > s1

Pour l’ensemble de sushis: {s1,s2,s3}

Etape 1: génération de tous les sous groupe

Extraction de plusieurs sous groupes, selon des descripteurs que nous avons choisis ici car nous ne pouvons pas encore savoir quels peuvent être exactement les descripteurs intéressants.

Tous les candidats en tant que sous-groupe, avec les descripteurs : ville, age, ville&age :

```
c1: ville=caen - p1,p5,p6
c2: ville=cheux - p2, p3
c3: ville=carpiquet - p4
c4: age=jeune - p2, p5
c5: age=moyen - p1, p4
c6: age=vieux - p3,p6
c7: ville=caen & age=jeune - p5
c8: ville=caen & age=moyen - p1
c9: ville=caen & age=vieux - p6
c10: ville=cheux & age=jeune - p2
c11: ville=cheux & age=moyen - {}
c12: ville=cheux & age=vieux - p3
c13: ville=carpiquet & age=jeune - {}
c14:  ville=carpiquet & age=moyen - p4
c15: ville=carpiquet & age=vieux - {}
```

Ici, on va sélectionner les sous groupes qui ont plus d’un individu :
c1, c2, c4, c5, c6

Etape 2: Matrice de préférence de chacun des individus :

Matrice pour P1:

	s1	s2	s3
s1	0	1	1
s2	-1	0	-1
s3	-1	1	0

Matrice pour P2:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	1
s3	1	-1	0

Matrice pour P3:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	-1
s3	1	1	0

Matrice pour P4:

	s1	s2	s3
s1	0	1	1
s2	-1	0	1
s3	-1	-1	0

Matrice pour P5:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	1
s3	1	-1	0

Matrice pour P6:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	-1
s3	1	1	0

Étape 3 : Matrice de préférence moyenne de la population “MD”

Préférence	P1	P2	P3	P4	P5	P6	Mean
s1 - s2 :	1	-1	-1	1	-1	-1	-0.33
s1 - s3 :	1	-1	-1	1	-1	-1	-0.33
s2 - s1:	-1	1	1	-1	1	1	0.33
s2 - s3 :	-1	1	-1	1	1	-1	0
s3 - s1 :	-1	1	1	-1	1	1	0.33
s3 - s2 :	1	-1	1	-1	-1	1	0

Matrice MD:

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

Etape 4: Matrices moyenne de préférence pour chacun des sous groupes :

c1: p1,p5,p6

Préférence	P1	P5	P6	Mean
s1 - s2 :	1	-1	-1	-0.33
s1 - s3 :	1	-1	-1	-0.33
s2 - s1:	-1	1	1	0.33
s2 - s3 :	-1	1	-1	-0.33
s3 - s1 :	-1	1	1	-0.33
s3 - s2 :	1	-1	1	0.33

Matrice Ms pour c1:

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	-0.33
s3	0.33	0.33	0

c2: p2,p3

Préférence	P2	P3	Mean
s1 - s2 :	-1	-1	-1
s1 - s3 :	-1	-1	-1
s2 - s1:	1	1	1
s2 - s3 :	1	-1	0
s3 - s1 :	1	1	1
s3 - s2 :	-1	1	0

Matrice Ms pour c2:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	0
s3	1	0	0

c4: p2,p5

Préférence	P2	P5	Mean
s1 - s2 :	-1	-1	-1
s1 - s3 :	-1	-1	-1
s2 - s1:	1	1	1
s2 - s3 :	1	1	1
s3 - s1 :	1	1	1
s3 - s2 :	-1	-1	-1

Matrice Ms pour c4:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	1
s3	1	-1	0

c5: p1,p4

Préférence	P1	P4	Mean
s1 - s2 :	1	1	1
s1 - s3 :	1	1	1
s2 - s1:	-1	-1	-1
s2 - s3 :	-1	1	0
s3 - s1 :	-1	-1	-1
s3 - s2 :	1	-1	0

Matrice Ms pour c5:

	s1	s2	s3
s1	0	1	1
s2	-1	0	0
s3	-1	0	0

c6: p3,p6

Préférence	P3	P6	Mean
s1 - s2 :	-1	-1	-1
s1 - s3 :	-1	-1	-1
s2 - s1:	1	1	1
s2 - s3 :	-1	-1	-1
s3 - s1 :	1	1	1
s3 - s2 :	1	1	1

Matrice Ms pour c6:

	s1	s2	s3
s1	0	-1	-1
s2	1	0	-1
s3	1	1	0

Etape 5: application de la déviation

LS = ½ * (MD - Ms)

sur c1: LSc1 =

½ * (

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

-

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	-0.33
s3	0.33	0.33	0

=

½*

0	0	0
0	0	0.33
0	-0.33	0

= LSc1

0	0	0
0	0	0.165
0	-0.165	0

sur c2: LSc2 =

½ * (

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

-

	s1	s2	s3
s1	0	-1	-1
s2	1	0	0
s3	1	0	0

)

=

½*

0	0.67	0.67
-0.67	0	0
-0.67	0	0

=

LSc2

0	0.335	0.335
-0.335	0	0
-0.335	0	0

sur c4: LSc4 =

½ * (

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

-

	s1	s2	s3
s1	0	-1	-1
s2	1	0	1
s3	1	-1	0

)

=

½*

0	0.67	0.67
-0.67	0	-1
-0.67	1	0

=

LSc4

0	0.335	0.335
-0.335	0	-0.5
-0.335	0.5	0

sur c5: LSc5 =

½ * (

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

-

	s1	s2	s3
s1	0	1	1
s2	-1	0	0
s3	-1	0	0

)

=

½*

0	-1.33	-1.33
1.33	0	0
1.33	0	0

=

LSc5

0	-0.665	-0.665
0.665	0	0
0.665	0	0

sur c6: LSc6 =

½ * (

	s1	s2	s3
s1	0	-0.33	-0.33
s2	0.33	0	0
s3	0.33	0	0

-

	s1	s2	s3
s1	0	-1	-1
s2	1	0	-1
s3	1	1	0

)

=

½*

0	0.67	0.67
-0.67	0	1
0.67	-1	0

=

LSc6

0	0.335	0.335
-0.335	0	0.5
-0.335	-0.5	0

Etape 6 : Récupération de sous groupes intéressants via les mesures de qualité :

$\sqrt{s/n}$: s= taille du sous groupe donc nombre individus du sous groupe , n= nombre d'individus total de la population étudiée. n=6 ici.

méthode de rankingWise Norme:

$$\mathrm{RWNorm}(S) = \sqrt{s/n} \cdot ||L_S||_F = \sqrt{s/n} \cdot \sqrt{\sum_{i=1}^{k} \sum_{j=1}^{k} L_S(i,j)^2}$$

rwnorm sur c1:

pour c1: s=3

$= \sqrt{3/6} * \sqrt{lsc1f}$:

(

0	0	0
0	0	0.165
0	-0.165	0

f)

$= \sqrt{3/6} * \sqrt{0.165^2 + (-0.165)^2}$

$= \sqrt{3/6} * \sqrt{0.5445}$

$= 0,521775814$

pour c2: s=2

$= \sqrt{2/6} * \sqrt{lsc2f}$:

(

0	0.335	0.335
-0.335	0	0
-0.335	0	0

f)

$= \sqrt{2/6} * \sqrt{0.335^2 + 0.335^2 + (-0.335)^2 + (-0.335)^2}$

$= \sqrt{2/6} * \sqrt{0,4489}$

$= 0,38682468$

pour c4: s=2

$= \sqrt{2/6} * \sqrt{lsc4f}$:

(

0	0.335	0.335
-0.335	0	-0.5
-0.335	0.5	0

f)

$= \sqrt{2/6} * \sqrt{0.335^4 + (-0.335)^4 + (-0.5)^2 + 0.5^2}$

$= \sqrt{2/6} * \sqrt{0,9489}$

$= 0,562405548$

pour c5: s=2

$= \sqrt{2/6} *$ $\sqrt{lsc5f}$:

(

0	-0.665	-0.665
0.665	0	0
0.665	0	0

f)

$= \sqrt{2/6} *$

$\sqrt{(-\ 0.665)^2 + (-\ 0.665)^2 + \ 0.665^2 + \ 0.665^2}$

$= \sqrt{2/6} * \ \sqrt{1,7689}$

$= \ 1,02127489$

pour c6: s=2

$= \sqrt{2/6} *$ $\sqrt{lsc6f}$:

(

0	0.335	0.335
-0.335	0	0.5
-0.335	-0.5	0

f)

$= \sqrt{2/6} *$

$\sqrt{0.335^4 + (-\ 0.335)^4 + 0.5^2 + \ (-\ 0.5)^2}$

$= \sqrt{2/6} * \ \sqrt{0,9489}$

$= \ 0,562405548$

On a donc :

`rwnormc1` $= 0,521775814$

`rwnormc2` $= 0,38682468$

`rwnormc4` $= 0,562405548$

`rwnormc5` $= \ 1,02127489$

`rwnormc6` $= 0,562405548$

Groupe(s) de ranking moyen très différent(s): c5 car c'est le plus haut, et le plus éloigné par rapport aux autres.

Méthode de Labelwise norme :

$$\mathrm{LWNorm}(S) = \sqrt{s/n} \cdot \max_{i=1,\ldots,k} \sqrt{\sum_{j=1}^{k} L_S(i,j)^2}$$

lsc1:

0	0	0
0	0	0.165
0	-0.165	0

lwnormc1 = $\sqrt{3/6}$* max racine de (0+0+0 ou 0+0+0.165^2 ou 0+(-0.165)^2)+0
= $\sqrt{3/6}$* racine0.165
= 0,116672619

lsc2:

0	0.335	0.335
-0.335	0	0
-0.335	0	0

lwnormc2 = $\sqrt{2/6}$* max (0+0.335^2+0.335^2 ou -0.335^2+0+0 ou -0.335^2+0+0
= $\sqrt{2/6}$* (racine de 0,22445 ou 0,112225)
= $\sqrt{2/6}$* $\sqrt{0,22445}$ = 0,273526355

lsc4:

0	0.335	0.335
-0.335	0	-0.5
-0.335	0.5	0

lwnormc4 = $\sqrt{2/6}$* max (0+0.335^2+0.335^2 ou -0.335^2+0+0.5^2 ou -0.335^2+0.5^2+0
= $\sqrt{2/6}$* (racine de 0,22445 ou 0,362225)
= $\sqrt{2/6}$* $\sqrt{0,362225}$
= 0,347479016

lsc5:

0	-0.665	-0.665
0.665	0	0
0.665	0	0

lwnormc5 = $\sqrt{2/6}$* max (0+(-0.665)^2+(-0.665)^2 ou (-0.665)^2+0+0 ou (-0.665)^2+0+0)
= $\sqrt{2/6}$* (racine de 0,88445 ou 0,442225)
= $\sqrt{2/6}$* $\sqrt{0,88445}$ = 0,542970226

lsc6:

0	0.335	0.335
-0.335	0	0.5
-0.335	-0.5	0

lwnormc6 = $\sqrt{2/6}$* max
(0+0.335^2+0.335^2 ou
(-0.335)^2+0+0.5^2 ou
(-0.335)^2+0.5^2+0)
= $\sqrt{2/6}$* (racine de 0,22445 ou 0,362225)
= $\sqrt{2/6}$* $\sqrt{0,362225}$ = 0,347479016

On a donc:
lwc1= 0,116672619
lwc2= 0,273526355
lwc4= 0,347479016
lwc5= 0,542970226
lwc6= 0,347479016

Il est dit dans l'article que les hautes valeurs montrent que le label se comporte différemment, ici c5 est très nettement supérieur aux autres.

Méthode de Pairwise :

$$\text{PWMax}(S) = \sqrt{s/n} \cdot \max_{i,j=1,\ldots,k} |L_S(i,j)|$$

pwc1 = $\sqrt{3/6}$ * 0.165 = 0,116672619
pwc2 = $\sqrt{2/6}$ * 0.335 = 0,19341234
pwc4 = $\sqrt{2/6}$ * 0.335 = 0,19341234
pwc5 = $\sqrt{2/6}$ * 0.665 = 0,383937929
pwc6 = $\sqrt{2/6}$ * 0.5 = 0,288675135

Le sous groupe c5 est encore une fois très haut.

Récapitulatif :
En ayant fait toutes ces mesures, on peut dire que c5 est un groupe qui se comporte différemment par rapport à l'ensemble du groupe de 6 personnes.
On sait que c5 est composé de personnes d'âge moyen, P1 et P4,
De manière moyenne(Msc5), ces 2 personnes préfèrent le sushi numéro 1 contre tous les autres sushis.
Alors qu' en moyenne, l'ensemble de la population aime le moins le sushi numéro 1.
Le sous-groupe c5 est donc bien un groupe intéressant.