ISBS2 / 2016-2017 **Données Massives**

TD – Data Analysis
clustering

Introduction

Vous disposez d'un fichier compilant données cliniques et annotations fonctionnelles obtenues par l'analyse métagénomique des selles des patients. L'annotation fonctionnelle (notée **AF** dans ce TD) utilisée ici se base sur la banque de données d'annotation **Interpro2go** ; elle organise les annotations en 3 sous-catégories :

"*biological process*" (BP)

"*cellular component*" (CC)

"*molecular function*" (MF)

Une annotation fonctionnelle (par exemple "*IPR005814 Aminotransferase class-III*") est exprimée sous la forme d'une valeur entre 0 et 100, que l'on peut interpréter comme la proportion relative de cette fonction (par rapport à l'ensemble des autres fonctions) dans le métagenome. Notez qu'une fonction est identifiée à partir des gènes connus pour coder cette fonction, et donc cela ne présuppose en rien de quelle bactérie porte cette fonction. Il faut donc plutôt voir cette annotation fonctionnelle comme un **potentiel** de la communauté bactérienne intestinale du patient.

Dans ce TD, la question est de savoir si le potentiel fonctionnel pourrait caractériser les patients.

Votre groupe va se concentrer sur une des trois sous-catégories, donné pour un niveau de détail plus fin que le TD précédent : par exemple BP4, CC4 ou MF4.

Attention, ces analyses nécessitent de nombreux essais/erreurs, et de bien regarder les données (donc la data exploration) et d'entrer de nombreuses commandes R (utilisez le rappel des touches et/ou l'historique). N'espérez pas que de l'information va surgir en cliquant 2 ou 3 fois…. Le "*data mining*" n'est pas de la sorcellerie, ni une "*roue de la chance*". Bref c'est une discipline scientifique...

Objectif de l'analyse

Vous allez regrouper les annotations fonctionnelles en cluster, puis voir si ces clusters sont représentatifs d'un sous-groupe de patients dans la cohorte.

Voici des idées de questions à aborder :

- combien d'AF ont été identifiées dans votre catégorie au niveaul 4 ?
- quels sont les intervalles de variation des AF ?
- Pouvez-vous exclure des AF que l'on peut considérer comme "peu importantes" (en terme de représentativité) ? Ceci permettrait de restreindre le nombre d'AF à clusteriser.
- Quels clustering obtenez-vous ? Combien de cluster allez-vous considérer ? Quelle distance, quelle méthode donne le "meilleur" clustering hiérachique et/ou k-means.
- Idéalement, vous ne conservez qu'un clustering. Peut-être deux…

Une fois cette étape effectuée, la question biologique que vous devez addresser est : “*ces clusters sont-ils discriminant ?*”. En d’autres termes, y a-t-il des variables cliniques qui sont “*très différentes*” entres les clusters (homme/femme, jeunes/vieux, témoins/CCR, etc.) ?

Si oui, c’est un premier résultat intéressant qui pourrait orienter vers un certain profil fonctionnel et donc l’importance à prendre en considération ce dernier.

Si non, c’est un premier résultat intéressant, car personne n’avait testé ce phénomène.

Petit balade pour bien commencer.

Je charge mon jeu de données et je m’assure qu’il est correctement lu :

```
> m = read.csv("EC2M3-WGS-GO-MF3_dataset-norm.csv", header=T, sep=";" )
> dim(m)
```

Maintenant, je peux regarder les noms des colonnes, voir celles qui caractérisent l’AF et voir quelles AF sont très peu représentées :

```
> colnames(m)
> summary( m[,55:75] )
> colMeans( m[,55:75] ) < 0.1
```

La fonction *colMean()* calcule la moyenne sur chaque colonne d’une matrice ou d’un data.frame. Je teste si cette moyenne est inférieure à un seul, ici 0.1%

```
> sum( colMeans( m[,55:75] ) < 0.1 ) # combien ?
> which( colMeans( m[,55:75] ) < 0.1 ) # lesquels ?
```

En première approche, je ne vais pas considérer ces AF. Je duplique mon data.frame et j’enlève les colonnes des AF que je considère comme pas assez informatives :

```
> mm = m[ , -( L + 55 ) ]
```

Attention, ici je dois rajouter 55 car la position que mon retourne la fonction *which()* est par rapport à mon bloc 55:75. Donc une valeur de 2 signifie le 2-ième colonnes dans le bloc, donc en position 55+2 dans mon data.frame *m*.

Maintenant j’extrais les valeurs d’AF pour commencer mon clustering. Je stocke ces valeurs dans un data.frame :

```
> maf = mm[ , 55:70 ]
> is.matrix(maf)
> is.data.frame(maf)
> dim(maf)
Je peux donc commencer le clustering :
> dij = dist( maf )
> image( as.matrix(dij) )
```

Bon, la visualisation des distances 2 à 2 ne semble pas très informatif. Peut-être faudrait-il jouer surla palette de couleurs. Ca valait le coup de regard.

```
> hc = hclust( dij )
> plot( hc )
```

Voici mon premier dendrogramme. Voyons comment le “couper” en sous-clusters.

```
> rect.hclust( hc, k=2 )
> plot( hc ) # deux commandes rect.hclust successives se superposent.
> rect.hclust( hc, k=5 )
> plot( hc )
> rect.hclust( hc, k=6 )
```

Ok, supposons ici k=6.

```
> gp = cutree( hc, k=6 )
> table(gp)
gp
 1  2  3  4  5  6
59 58  1 21 11  6
>
```

Voici la répartition de ma cohorte : on a ici deux groupes plus importants en terme de taille (59 et 58 individus), un singleton et des petits groupes.

Voyons quelques variables cliniques de ces sous-groupes d'individus (→ sous-groupe d'individu = un des 6 clusters). Je procède en deux fois : d'abord attribution à chaque individu de son cluster dans le data.frame, puis exploration des données.

```
> mc = m # je duplique mon data.frame initiale car je sais que je vais
beaucoup changer les choses dans ma session
> mc$gp = gp
# ou bien : mc = cbind( m, gp )> boxplot( m$Age ~ m$gp )
> is.factor( m$gp )
[1] FALSE
> m$gp = as.factor( m$gp )
> is.factor( m$gp )
[1] TRUE
```

Et finalement les statistiques descriptives :

```
> by( m$Age, m$gp, mean )
```

Ah peut-être une petite différence d'age entre les cluster 1 & 2 (les deux plus importants en taille). Voyons graphiquement :

```
> boxplot( m$Age ~ m$gp )
```

Eh non, finalement un écart-type est une donnée importante non ?

Voyons l'autre méthode de clustering :

```
> km = kmeans( maf, centers=6, nstart=25 )
> km
> # j'ajoute dans mon data.frame :
> m$km = km$cluster
# Y a-t-il un effet sur l'age ?
> boxplot( m$Age ~ m$km )
```

Maintenant à vous de jouer.

Pour ceux qui auront le package “factoextra”

Voici quelques commandes qui offrent très rapidement une visualisation plus agréable et des tests de valeurs optimale de *k.*

```
> install.packages("factoextra")
> library(factoextra)
> library(cluster)

> maf = mm[ , 55:70 ]
> dij = get_dist( maf )
> fviz_dist( dij )
```

Et maintenant un k-means :

```
> k6 = kmeans( maf, centers=6, nstart=25 )
> fviz_cluster(k6, data = maf)
```

Testons la “meilleure” valeur de *k* suivant quelques statistiques :

```
> fviz_nbclust(maf, kmeans, method = "wss")
> fviz_nbclust( maf, kmeans, method = "silhouette")
> gap_stat <- clusGap( maf, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
> fviz_gap_stat(gap_stat)
```

Les résultats ne sont pas cohérents entre eux, ce qui illustre bien la difficulté de la tache et le fait qu’une mesure seule ne permet pas de décider de la valeur de k.

Et en tri hiérarchique :

```
> hc = hclust( dij )
> plot( hc )
> rect.hclust( hc, k=6, border= 2:5 )
> g6 = cutree( hc, k=6 ) # sous-groupes obtenus avec 6 clusters
> fviz_cluster(list(data = maf, cluster = g6))
```

Attention dans le dernier exemple, ici il faut donner le data.frame qui a servi pour les calculs de distance, et les sous-groupes obtenus.

Rédaction du rapport

Votre rapport doit comprendre un partie “data exploration” des données cliniques (normalement faite au TD1) et des AF, une partie “clustering” (comment avez-vous sélectionné votre clustering) puis une partie “*relation entre données cliniques et clusters*”.

Revoici les variables cliniques que vous pouvez explorer :

```
Age; Gender, BMI, FOBT, WIF1, Group, Poids, Taille
éventuellement : leptin_moy, NPY_moy, Ghrelin_moy
```