Cluster Analysis with Python

HAC and K-Means

Ricco.Rakotomalala
http://eric.univ-lyon2.fr/~ricco/cours

Data importation, descriptive statistics

DATASET

Goal of study
Clustering of cheese dataset

Goal of the study

This tutorial describes a cluster analysis process. We deal with a set of cheeses (29 instances) characterized by their nutritional properties (9 variables). The aim is to determine groups of homogeneous cheeses in view of their properties.

We inspect and test two approaches using two Python procedures: the Hierarchical Agglomerative Clustering algorithm (SciPy package) ; and the K-Means algorithm (scikit-learn package).

The data file "fromage.txt" comes from the teaching page of Marie Chavent from the University of Bordeaux. The excellent course materials and corrected exercises (commented R code) available on its website will complete this tutorial, which is intended firstly as a simple guide for the introduction of the R software in the context of the cluster analysis.

Processing tasks

- Importing the dataset. Descriptive statistics.
- Cluster analysis – HAC and K-Means
- Potential solutions for determining the number of clusters
- Description and interpretation of the clusters

Cheese dataset

Fromages	calories	sodium	calcium	lipides	retinol	folates	proteines	cholesterol	magnesium
CarredelEst	314	353.5	72.6	26.3	51.6	30.3	21	70	20
Babybel	314	238	209.8	25.1	63.7	6.4	22.6	70	27
Beaufort	401	112	259.4	33.3	54.9	1.2	26.6	120	41
Bleu	342	336	211.1	28.9	37.1	27.5	20.2	90	27
Camembert	264	314	215.9	19.5	103	36.4	23.4	60	20
Cantal	367	256	264	28.8	48.8	5.7	23	90	30
Chabichou	344	192	87.2	27.9	90.1	36.3	19.5	80	36
Chaource	292	276	132.9	25.4	116.4	32.5	17.8	70	25
Cheddar	406	172	182.3	32.5	76.4	4.9	26	110	28
Comte	399	92	220.5	32.4	55.9	1.3	29.2	120	51
Coulomniers	308	222	79.2	25.6	63.6	21.1	20.5	80	13
Edam	327	148	272.2	24.7	65.7	5.5	24.7	80	44
Emmental	378	60	308.2	29.4	56.3	2.4	29.4	110	45
Fr.chevrepatemolle	206	160	72.8	18.5	150.5	31	11.1	50	16
Fr.fondu.45	292	390	168.5	24	77.4	5.5	16.8	70	20
Fr.frais20nat.	80	41	146.3	3.5	50	20	8.3	10	11
Fr.frais40nat.	115	25	94.8	7.8	64.3	22.6	7	30	10
Maroilles	338	311	236.7	29.1	46.7	3.6	20.4	90	40
Morbier	347	285	219	29.5	57.6	5.8	23.6	80	30
Parmesan	381	240	334.6	27.5	90	5.2	35.7	80	46
Petitsuisse40	142	22	78.2	10.4	63.4	20.4	9.4	20	10
PontlEveque	300	223	156.7	23.4	53	4	21.1	70	22
Pyrenees	355	232	178.9	28	51.5	6.8	22.4	90	25
Reblochon	309	272	202.3	24.6	73.1	8.1	19.7	80	30
Rocquefort	370	432	162	31.2	83.5	13.3	18.7	100	25
SaintPaulin	298	205	261	23.3	60.4	6.7	23.3	70	26
Tome	321	252	125.5	27.3	62.3	6.2	21.8	80	20
Vacherin	321	140	218	29.3	49.2	3.7	17.6	80	30
Yaourtlaitent.nat.	70	91	215.7	3.4	42.9	2.9	4.1	13	14

Row names

Active variables

Data file

Data importation, descriptive statistics and plotting

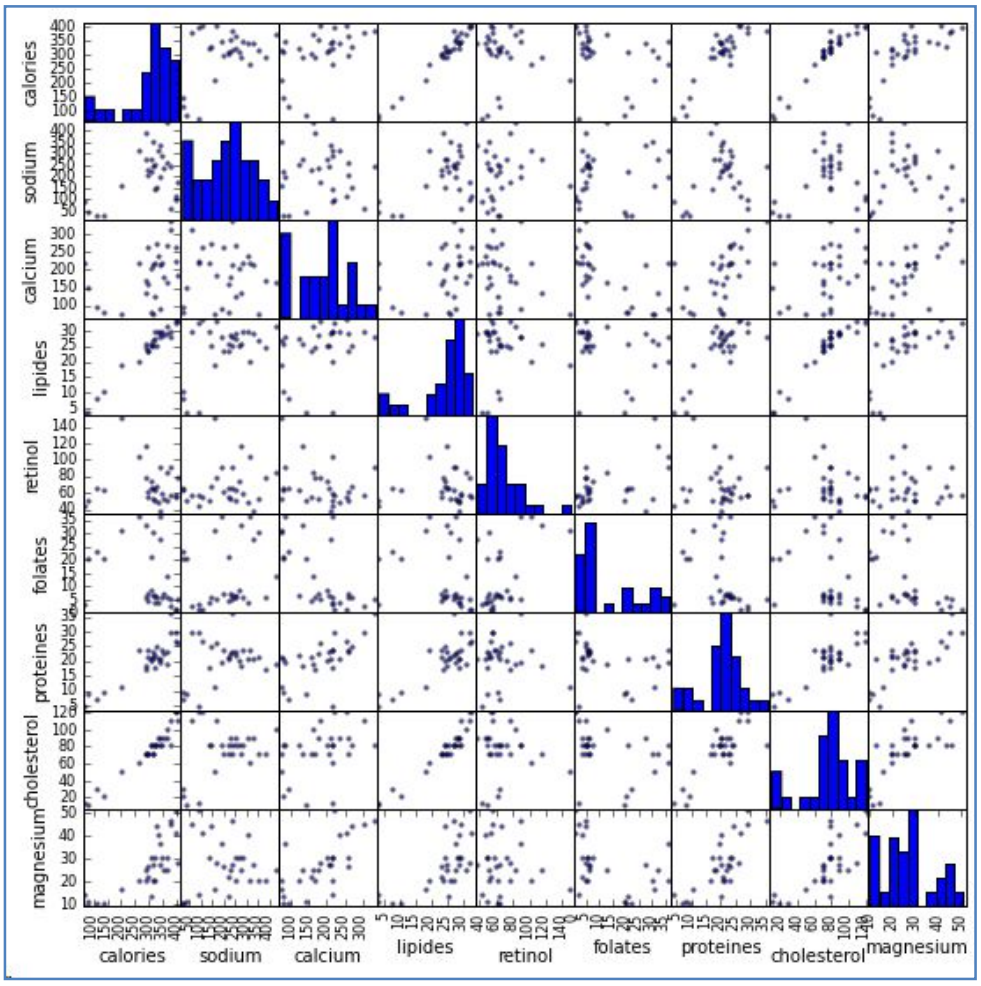
```
#modifying the default working directory
import os
os.chdir("…")

#loading the dataset
import pandas
fromage = pandas.read_table("fromage.txt",sep="\t",header=0,index_col=0)

#dimension of the dataset (number of rows and columns)
print(fromage.shape)

#descriptive statistics
print(fromage.describe())

#pairwise scatterplots
from pandas.tools.plotting import scatter_matrix
scatter_matrix(fromage,figsize=(9,9))
```



This kind of graph is never trivial. For instance, we note that (1) "lipides" is highly correlated to "calories" and "cholesterol"(*this is not really surprising, but it means also that the same phenomenon will weigh 3 times more in the study*) ; (2) in some situations, some groups seem naturally appeared (*e.g. "proteines" vs. "cholesterol", we identify a group in the southwest of the scatterplot, with high inter-groups correlation*).

Hierarchical Agglomerative Clustering

HAC

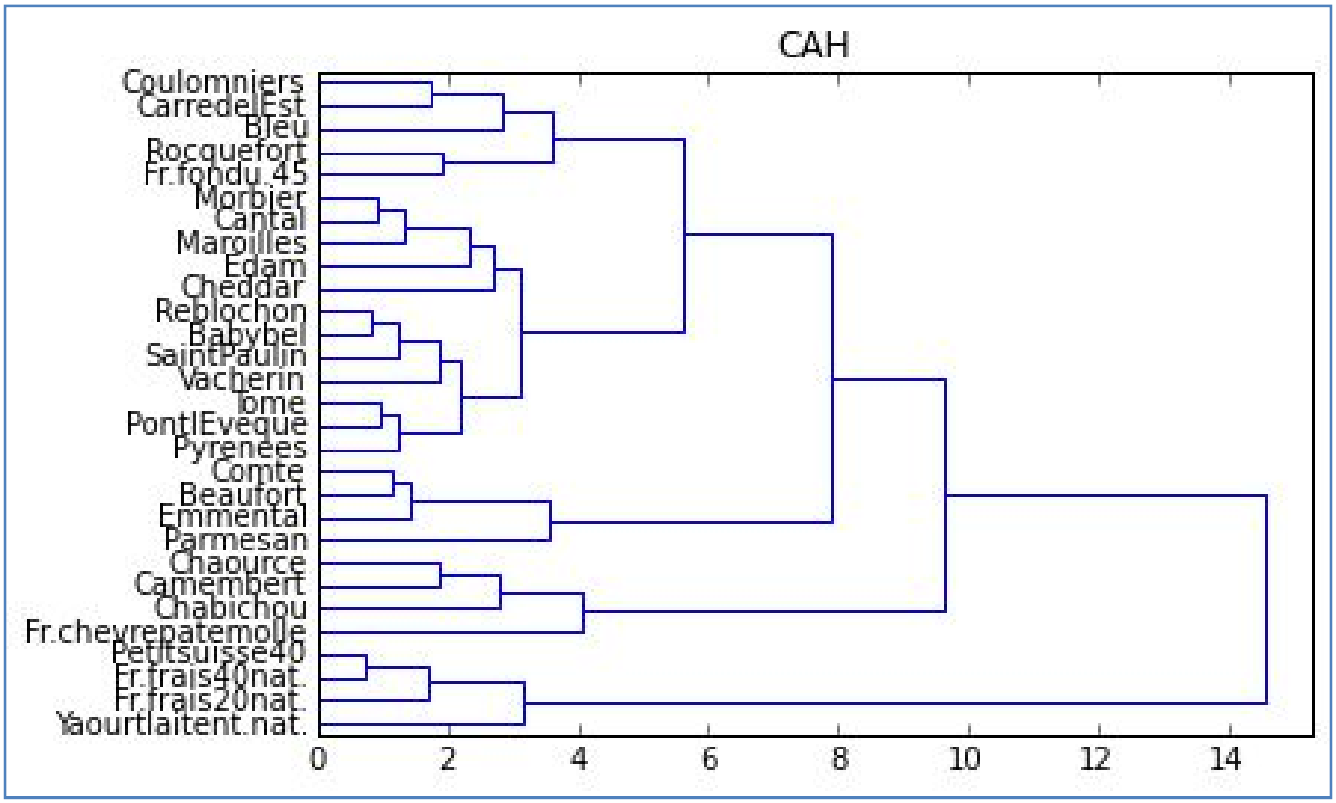
Hierarchical Agglomerative Clustering

Using the "**scipy**" package

```
#libraries for plotting and the HAC
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage

#generate the linkage matrix – Ward method
Z = linkage(fromage_cr,method='ward',metric='euclidean')

#plotting the dendrogram
plt.title("CAH")
dendrogram(Z,labels=fromage.index,orientation='left',color_threshold=0)
plt.show()
```



The dendrogram suggests a partitioning in 4 groups. It is noted that a group of cheeses, the "fresh Cheeses" (far left), seems very different to the others, to the point that we could have considered also a partitioning in 2 groups only. We will discuss this dimension longer when we combine the study with a principal component analysis (PCA).

Hierarchical Agglomerative Clustering

Partitioning into clusters - Visualization of the clusters

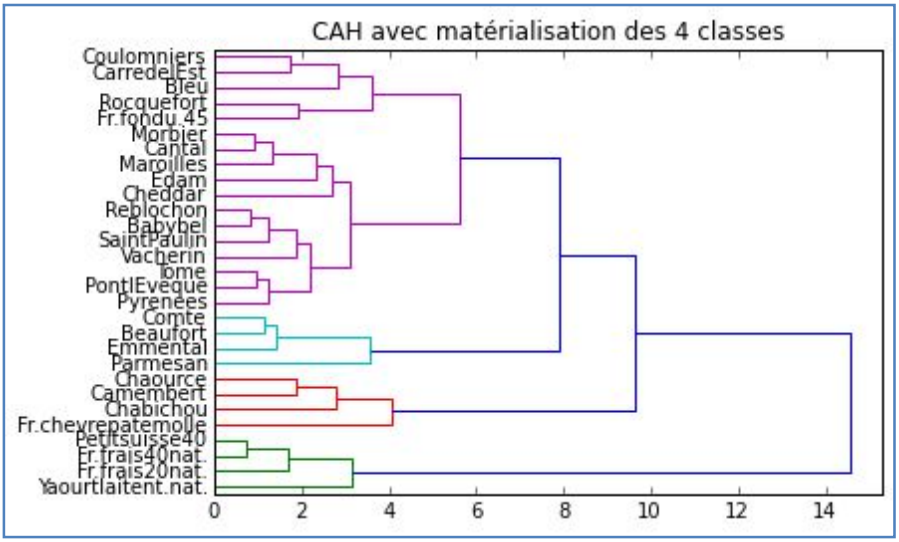
```
#highlighting of the 4 groups (height t = 7)
plt.title('CAH avec matérialisation des 4 classes')
dendrogram(Z,labels=fromage.index,orientation='left',color_threshold=7)
plt.show()

#cutting at the height t = 7 ==> cluster membership of cases
groupes_cah = fcluster(Z,t=7,criterion='distance')
print(groupes_cah)

#sorted index of the clusters
import numpy as np
idg = np.argsort(groupes_cah)

#printing the instance names and their cluster number
print(pandas.DataFrame(fromage.index[idg],groupes_cah[idg]))
```

Groupe	Fromage
1	Yaourtlaitent.nat.
1	Fr.frais20nat.
1	Petitsuisse40
1	Fr.frais40nat.
2	Fr.chevrepatemolle
2	Camembert
2	Chabichou
2	Chaource
3	Emmental
3	Parmesan
3	Beaufort
3	Comte
4	Pyrenees
4	PontlEveque
4	Rocquefort
4	SaintPaulin
4	Tome
4	Reblochon
4	CarredelEst
4	Maroilles
4	Vacherin
4	Edam
4	Coulomniers
4	Cheddar
4	Cantal
4	Bleu
4	Babybel
4	Morbier
4	Fr.fondu.45



The 4th group corresponds to the “fresh cheeses”.
The 3rh to the “soft cheeses”.
The 2nd to the “hard cheeses”.
The 1st is a bit the “catch all” group.

My skills about cheese stop there (thanks to Wikipedia).
For characterization using the variables, it is necessary to go through univariate (easy to read and interpret) or multivariate statistical techniques (which take into account the relationships between variables).

K-Means Clustering

K-MEANS

K-Means clustering

Using the "**scikit-learn**" package

```
#k-means on the standardized dataset
from sklearn import cluster
kmeans = cluster.KMeans(n_clusters=4)
kmeans.fit(fromage_cr)

#sorted index of the clusters
idk = np.argsort(kmeans.labels_)

#printing the instance names and their cluster number
print(pandas.DataFrame(fromage.index[idk],kmeans.labels_[idk]))

#distances to the centroids for each instance
print(kmeans.transform(fromage_cr))

#equivalences with the HAC clusters
pandas.crosstab(groupes_cah,kmeans.labels_)
```

Groupe	0	1	2	3
CarredelEst	2.22	5.53	5.22	2.92
Babybel	3.02	5.19	2.79	0.74
Beaufort	5.16	7.51	1.15	2.86
Bleu	3.24	6.12	3.90	2.11
Camembert	1.93	5.40	5.10	3.54
Cantal	4.02	6.30	2.20	1.19
Chabichou	1.78	5.93	4.53	3.39
Chaource	1.03	5.55	5.09	3.46
Cheddar	3.75	6.82	2.29	1.95
Comte	5.44	7.84	1.35	3.42
Coulomniers	1.96	4.84	4.75	2.49
Edam	4.23	6.13	1.34	2.25
Emmental	5.53	7.48	0.90	3.40
Fr.chevrepatemolle	3.10	5.01	7.09	5.54
Fr.fondu.45	2.84	5.28	4.45	1.89
Fr.frais20nat.	5.33	0.68	7.61	6.00
Fr.frais40nat.	4.55	1.01	7.32	5.61
Maroilles	4.20	6.46	2.45	1.53
Morbier	3.47	6.06	2.50	0.65
Parmesan	5.23	7.94	2.11	3.60
Petitsuisse40	4.38	1.21	7.13	5.40
PontlEveque	3.08	4.69	3.52	1.26
Pyrenees	3.28	5.71	2.81	0.71
Reblochon	2.74	5.31	2.94	0.81
Rocquefort	3.02	6.81	4.22	2.18
SaintPaulin	3.47	5.12	2.67	1.37
Tome	2.73	5.25	3.67	1.26
Vacherin	3.79	5.27	2.63	1.50
Yaourtlaitent.nat.	6.09	1.93	7.46	5.94

Distances to the centroids for each instance. The minimum value is highlighted.

Cluster membership

Classe	Fromages
0	CarredelEst
0	Camembert
0	Fr.chevrepatemolle
0	Chabichou
0	Chaource
0	Coulomniers
1	Petitsuisse40
1	Fr.frais40nat.
1	Fr.frais20nat.
1	Yaourtlaitent.nat.
2	Parmesan
2	Edam
2	Emmental
2	Beaufort
2	Comte
3	Tome
3	SaintPaulin
3	Rocquefort
3	Reblochon
3	Pyrenees
3	PontlEveque
3	Cheddar
3	Morbier
3	Maroilles
3	Bleu
3	Vacherin
3	Cantal
3	Babybel
3	Fr.fondu.45

col_0	0	1	2	3
row_0				
1	0	4	0	0
2	4	0	0	0
3	0	0	4	0
4	2	0	1	14

Equivalences between HAC and k-Means

The 1st group of the HAC is equivalent to the 2nd group (n°1) of the K-Means. After that, there are some connections, but they are not exact.

K-Means Algorithm
Determining the number of clusters

K-Means, unlike the CAH, does not provide a tool to help us to detect the number of clusters. We have to program them under R or use procedures provided by dedicated packages. The approach is often the same: we vary the number of groups, and we observe the evolution of an indicator of quality of the partition.

Below we calculate the silhouette index (overall average silhouette) according to the number of clusters. The solution corresponds to the number of clusters which maximizes the index.

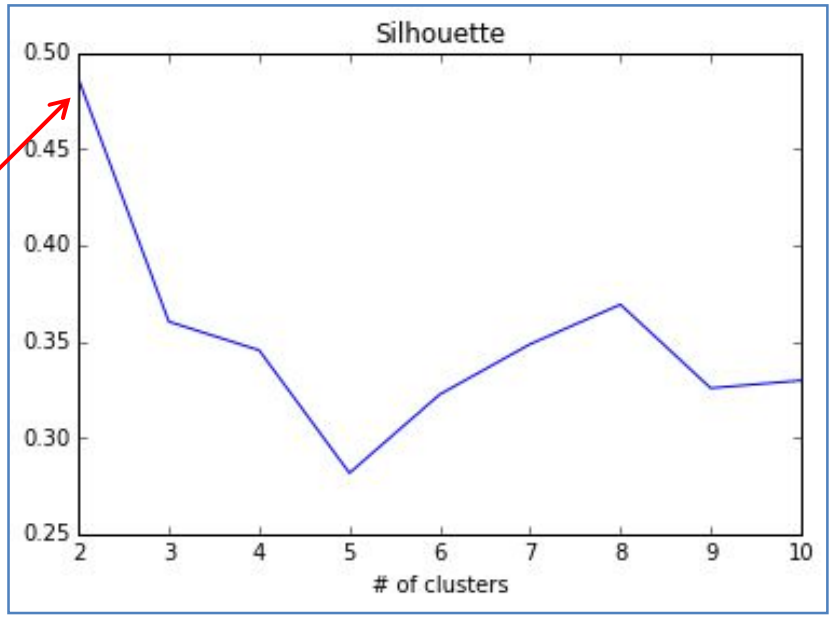
```
#library for evaluating the partitions
from sklearn import metrics

#measuring the "silhouette" score
#by varying the number of clusters from 2 to 10
res = np.arange(9,dtype="double")
for k in np.arange(9):
    km = cluster.KMeans(n_clusters=k+2)
    km.fit(fromage_cr)
    res[k] = metrics.silhouette_score(fromage_cr,km.labels_)

print(res)

#plotting nb. of clusters vs. silhouette score
import matplotlib.pyplot as plt
plt.title("Silhouette")
plt.xlabel("# of clusters")
plt.plot(np.arange(2,11,1),res)
plt.show()
```

The partitioning into k = 2 clusters is the best according to the silhouette score.



Conditional descriptive statistics and visualization

INTERPRETING THE CLUSTERS

Interpreting the clusters

Conditional descriptive statistics

The idea is to compare the means of the active variables conditionally to the groups. It is possible to quantify the overall amplitude of the differences with the proportion of explained variance (square of the correlation ratio). The process can be extended to auxiliary variables that was not included in the clustering process, but used for the interpretation of the results. For the categorical variables, we will compare the conditional frequencies. The approach is straightforward and the results easy to read. We should remember, however, that we do not take into account the relationship between the variables in this case (some variables may be highly correlated).

```
#overall mean for each variable
m = fromage.mean()
#TSS (total sum of squares)
TSS = fromage.shape[0]*fromage.var(ddof=0)
print(TSS)
#slicing the data.frame according to the groups
gb = fromage.groupby(kmeans.labels_)
#conditional groups size
nk = gb.size()
print(nk)
#conditional means
mk = gb.mean()
print(mk)
#(difference between cond. means and overall mean)²
EMk = (mk-m)**2
#weighted by the size of the groups
EM = EMk.multiply(nk,axis=0)
#sum => BSS (between sum of squares)
BSS = np.sum(EM,axis=0)
print(BSS)
#square of the correlation ratio
#proportion of variance explained
#for each variable
R2 = BSS/TSS
print(R2)
```

Clusters size

0	4
1	5
2	6
3	14

(Corr. Ratio)²

calories	0.863799
sodium	0.599117
calcium	0.620108
lipides	0.851983
retinol	0.382815
folates	0.760722
proteines	0.810316
cholesterol	0.797596
magnesium	0.796207

The definition of the groups is – above all – dominated by fat content (lipids, cholesterol and calories convey the same idea) and protein.

The group **n°o** is strongly determined by these variables, the conditional means are very different.

Conditional means

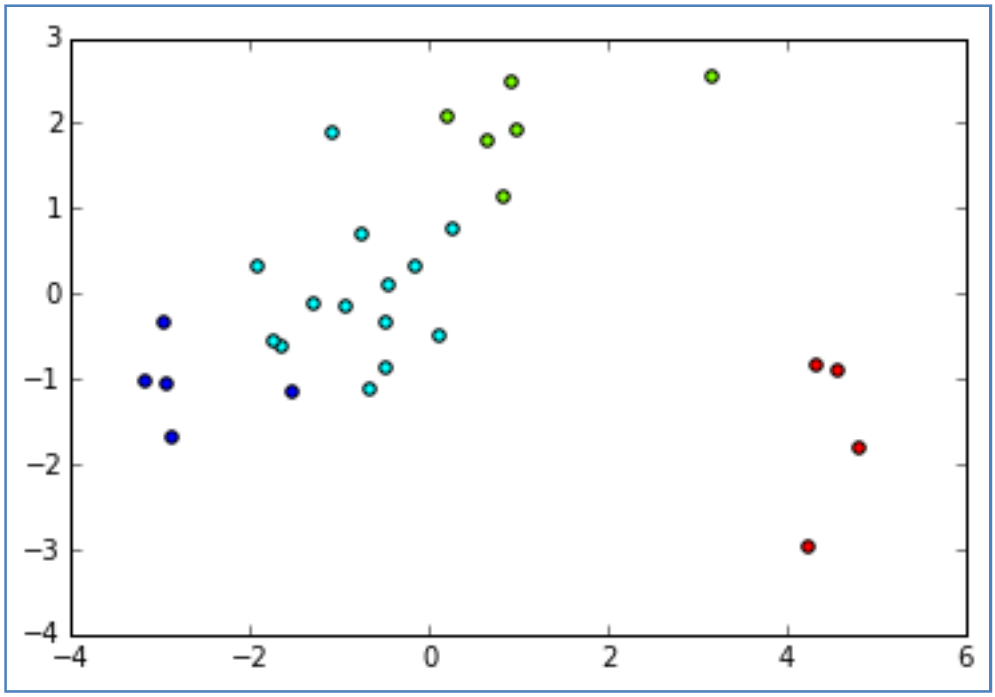
	calories	sodium	calcium	lipides	retinol	folates	proteines	cholesterol	magnesium
0	101.750000	44.750000	133.75	6.275000	55.150000	16.475000	7.200000	18.250000	11.250000
1	377.200000	130.400000	278.98	29.460000	64.560000	3.120000	29.120000	102.000000	45.400000
2	288.000000	252.916667	110.10	23.866667	95.866667	31.266667	18.883333	68.333333	21.666667
3	334.285714	267.428571	199.70	27.500000	60.050000	7.728571	21.228571	83.571429	27.142857

Interpreting the clusters
Principal component analysis (PCA)

When we combine the cluster analysis with factor analysis, we benefit from the data visualization to enhance the analysis. The main advantage is that we can take the relationship between the variables into account. But, on the other hand, we must also be able to read the outputs of the factor analysis correctly.

```
#PCA
from sklearn.decomposition import PCA
acp = PCA(n_components=2).fit_transform(fromage_cr)

#plotting the individuals into the factor map
#with various colors according to the cluster membership
#we note the role of the command zip() in the loop
for couleur,k in zip(['red','blue','lawngreen','aqua'],[0,1,2,3]):
   plt.scatter(acp[kmeans.labels_==k,0],acp[kmeans.labels_==k,1],c=couleur)
plt.show()
```



We note that there is a problem. The "fresh cheeses" group (group n°o in red) dominates the available information. The other cheeses are compressed into the left part of the scatter plot, making difficult to distinguish the other groups.

In the light of the results of PCA

COMPLEMENT THE ANALYSIS

Complement the analysis

Remove the "fresh cheeses" group from the dataset (1/2)

The fresh cheeses are so special – far from all the other observations – that they mask interesting relationships that may exist between the other products. We resume the analysis by excluding them from the treatments.

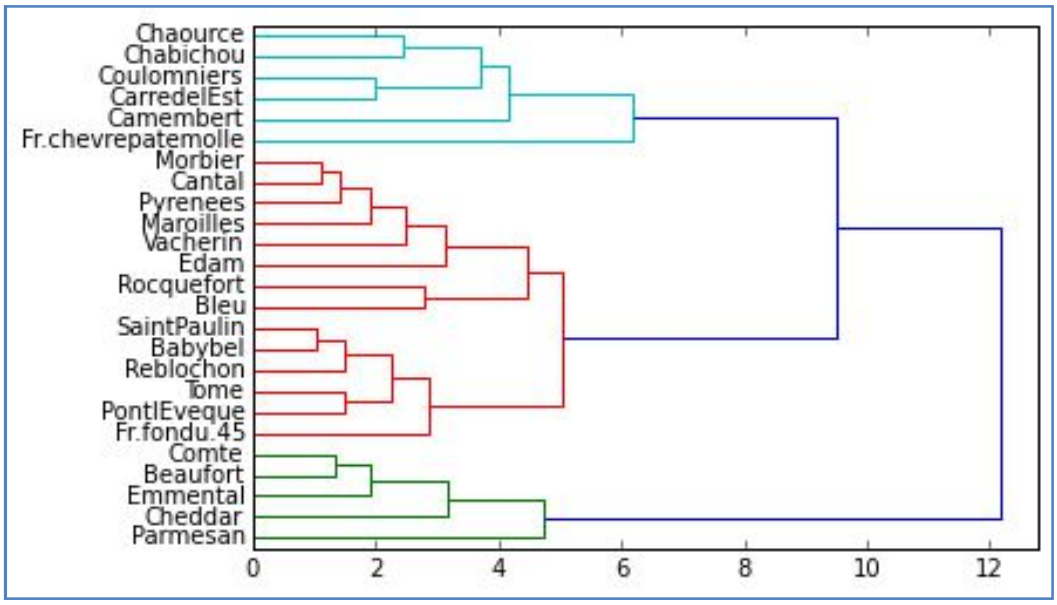
```
#remove the instances corresponding to the group n°0 from the k-means
fromage_subset = fromage.iloc[kmeans.labels_!=0,:]
print(fromage_subset.shape)

#standardize the new version of the dataset
fromage_subset_cr = preprocessing.scale(fromage_subset)

#generate the linkage matrix
Z_subset = linkage(fromage_subset_cr,method='ward',metric='euclidean')

#hac and plotting the dendrogram
plt.title("CAH")
dendrogram(Z_subset,labels=fromage_subset.index,orientation='left',color_threshold=7)
plt.show()

#groups
groupes_subset_cah = fcluster(Z_subset,t=7,criterion='distance')
print(groupes_subset_cah)
```



We can identify three groups. There is less the disrupting phenomenon observed in the previous analysis.

Complement the analysis

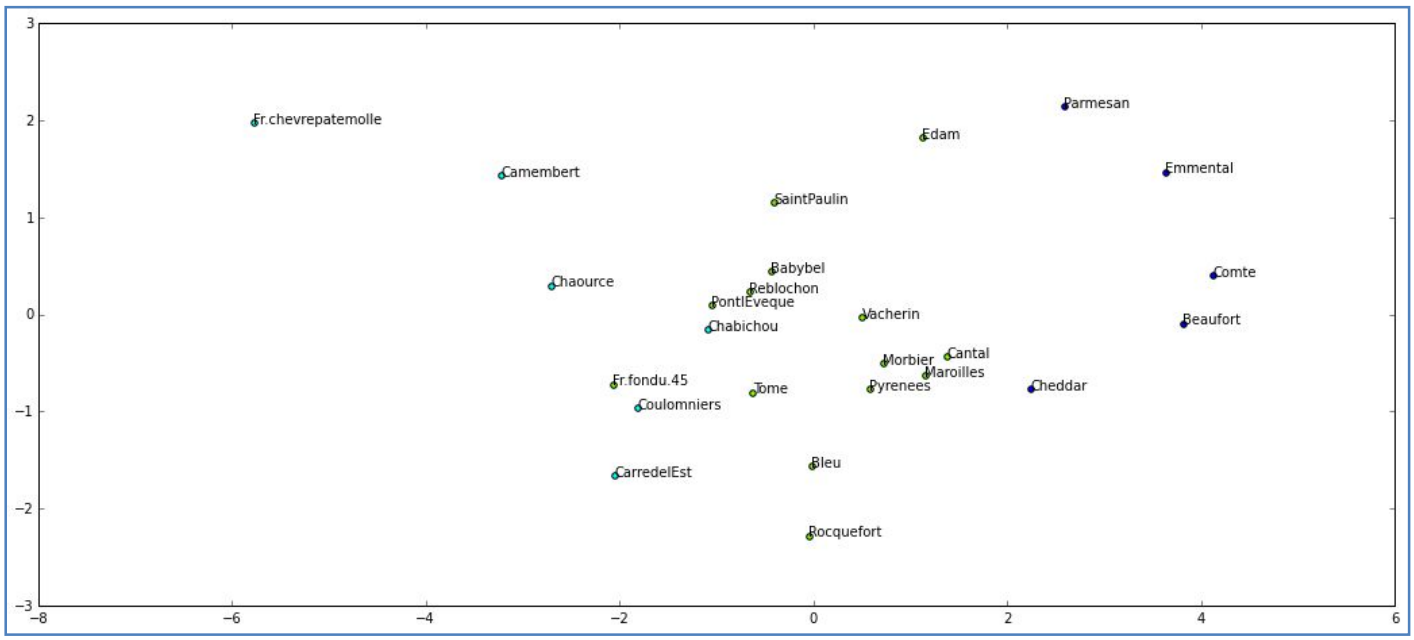
Remove the "fresh cheeses" group from the dataset (2/2)

```
#PCA
acp_subset = PCA(n_components=2).fit_transform(fromage_subset_cr)

#plotting the individuals into the factor map
#with various colors according to the cluster membership
#we note the role of the command zip() in the loop
plt.figure(figsize=(10,10))
for couleur,k in zip(['blue','lawngreen','aqua'],[1,2,3]):
            plt.scatter(acp_subset[groupes_subset_cah==k,0],acp_subset[groupes_subset_cah==k,1],c=couleur)

#set the row names into the scatter plot
#we note the role of enumerate() command
for i,label in enumerate(fromage_subset.index):
    plt.annotate(label,(acp_subset[i,0],acp_subset[i,1]))

plt.show()
```



The groups are mainly distinguishable on the first factor.

Some cheeses are assigned to other groups compared to the previous analysis

And we can do much more amazing things ...

References :

1. Chavent M. , Page de cours - Source of the « fromages.txt » dataset
2. Jörn's Blog, « SciPy Hierarchical Clustering and Dendrogram Tutorial ».
3. F. Pedregosa, G. Varoquaux, « Scikit-learn: machine learning in Python ».