



UNIVERSITÉ DE PARIS

Master 2 - Apprentissage Machine pour la Science des Données

Rapport de projet - Apprentissage non supervisé

Travail réalisé par :

- EL AMRANI Soumaya
- MOURABIT Nisrine
- MERBOUH Hajar

Contents

I.	Introduction	3
II.	Analyse Descriptive des Données	4
III.	Reduction de la dimension & Extraction des caractéristiques	7
3.1	Principal Components Analysis (PCA)	7
3.2	Réduction de la dimension basée sur les caractéristiques des séries temporelles	7
3.3	Transformation de Fourier	8
IV.	Clustering	9
4.1	Self-Organizing Maps (SOM)	9
4.2	K-means	11
4.3	Classification Hiérarchique (CAH)	13
4.3.1	Méthode directe	13
4.3.2	Méthode basé sur l'extraction des caractéristiques	14
4.4	Classification spectrale	15
V	Ruptures	17
5.1	Algorithme de programmation dynamique	17
5.2	Binary Segmentation Search Methods	18
VI	Conclusion	20

I. Introduction

La consommation électrique d'un foyer à un autre peut se trouver très différente. Cela est notamment dû au nombre d'équipements électroménagers et électriques du foyer qui se partage principalement entre quatre postes de dépenses : le chauffage, l'eau chaude, la cuisson et les autres appareils électriques, mais aussi à la fréquence d'utilisation de ces appareils et à la durée d'utilisation. L'électricité ne pouvant être stockée à grande échelle, l'équilibre entre la production et la consommation doit être maintenu en permanence afin d'assurer la sécurité du système et du réseau et d'éviter les pannes. Les comportements des consommateurs sont très variables et les regrouper en différents groupes homogènes peut être utile pour permettre une meilleure compréhension de leur consommation et proposer des offres plus adaptées ou pour prévoir la consommation de chaque groupe homogène et ainsi mieux comprendre la consommation globale en agrégeant ces prévisions.

L'objectif de ce projet est donc d'appliquer la classification non supervisée dans le cadre de la segmentation d'un ensemble de séries temporelles qui représentent la consommation d'électricité de 100 appartements/ménages observée toutes les 30 minutes durant 91 jours consécutifs. Il s'agit en particulier de détecter de manière automatique des ruptures (changements dans les habitudes de consommation) communes à l'ensemble des séries temporelles.



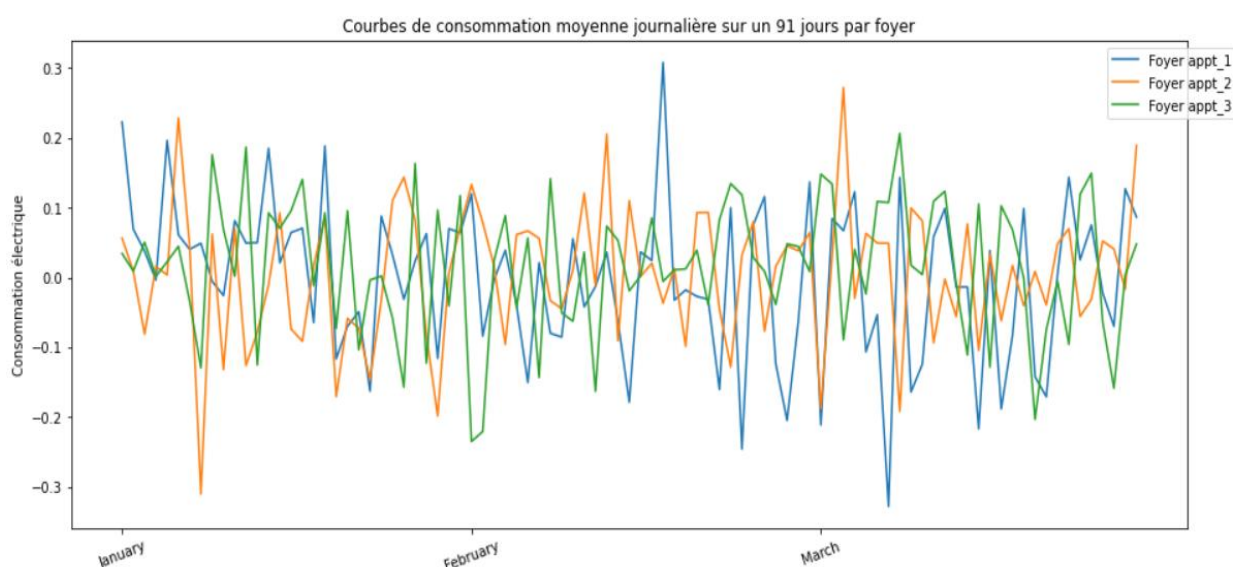
II. Analyse Descriptive des Données

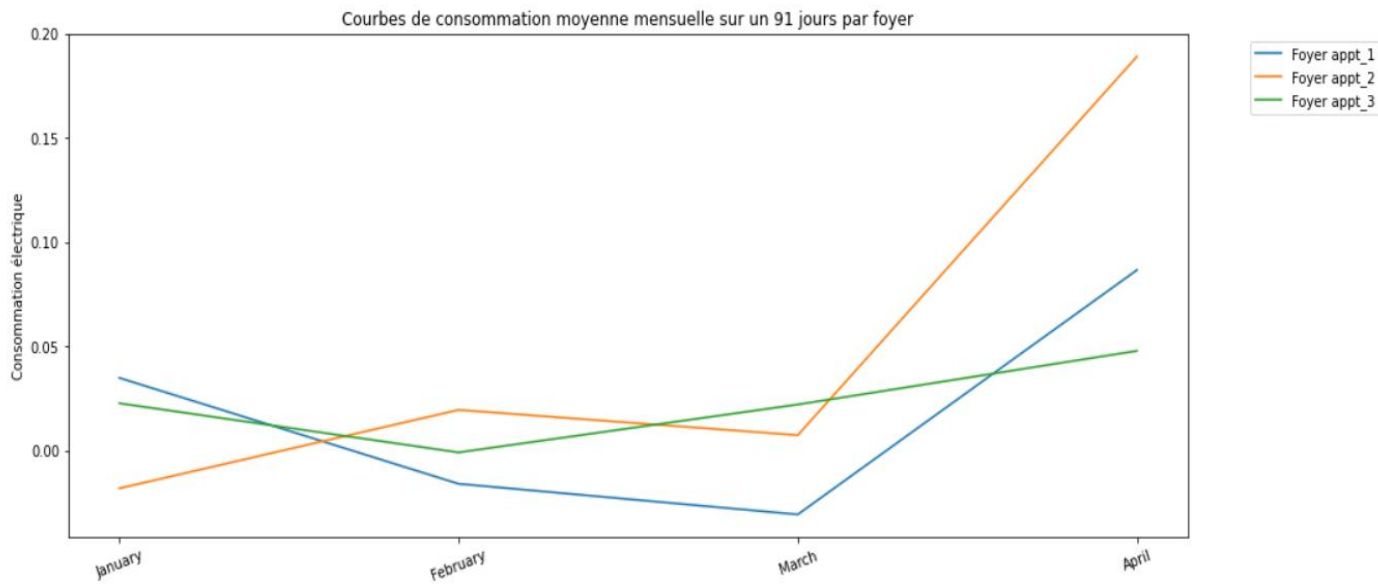
Nous ne voulons pour l'instant faire qu'une visualisation sommaire afin de mieux comprendre nos données, nous allons donc agréger les consommations par mois et par jour (Figure 1). L'avantage de ce type de visualisation réside dans le fait qu'elle nous permet d'observer facilement les différentes variations au niveau des comportements des ménages les plus sensibles, c'est-à-dire ceux dont la consommation d'énergie est la plus affectée par le facteur de temps : matin, soir, nuit, week-end et etc...

index	Day	appt_1	appt_2	appt_3	appt_4
0	01-01	10.69561212988773	2.7077605343697275	1.647497596815927	-5.9041530551950645
1	01-02	3.3177827065896377	0.4180418624906746	0.4435681954605485	3.8282572888989486
2	01-03	1.8167311858804107	-3.906144571585584	2.4258237306122257	-3.4631010864024154
3	01-04	-0.17094163062246698	0.7559874958664943	0.07030654566177841	-1.734171869704797
4	01-05	9.42707083144703	0.17068603863125564	1.0812961361293638	1.07806646389262

Figure 1 : Extrait des données après agrégation journalière

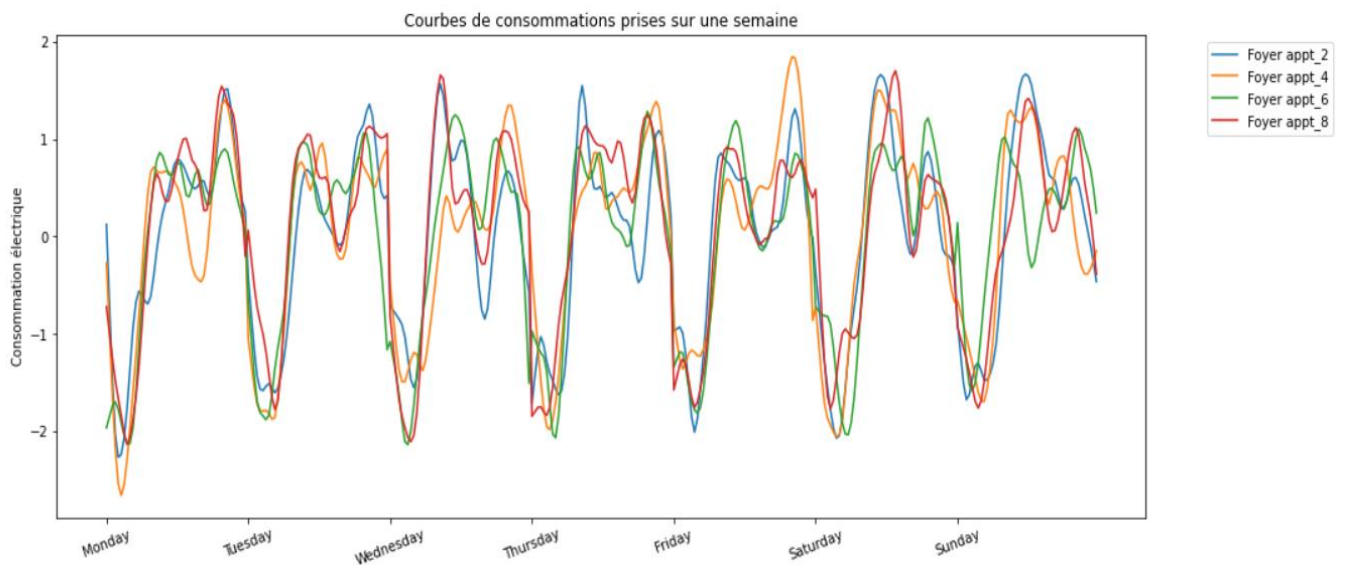
Nous pouvons maintenant représenter graphiquement les courbes de consommation par l'implémentation d'une fonction qui permet l'affichage des profils souhaités, en agréant par la moyenne des données individuelles à une échelle mensuelle et journalière.





L'agrégation des données à une échelle journalière semble écraser considérablement la variabilité des données, mais elle donne un aperçu des habitudes de consommation des différents ménages, et constitue une première technique d'agrégation des données.

Afin de pouvoir analyser des comportements plus fins et plus subtils, nous pouvons aussi visualiser des courbes représentant la consommation journalière moyenne de certains ménages pendant une semaine.



La figure ci-dessus nous permet de mieux comprendre le comportement des différents profils. Sur le graphique obtenu, celui qui montre la consommation sur une semaine donnée, les pics sont relativement élevés, on remarque notamment que les consommations des 4 foyers varie beaucoup. Le graphique semble suggérer ici que appt_8 mange le lundi et jeudi vers midi chez lui comme il utilise des appareils ménagers, tandis que appt_4 ne semble rentrer chez lui que le soir, bien qu'il consomme de l'électricité beaucoup plus le soir. Il semble aussi que

appt_6 consomme de l'énergie électrique beaucoup plus le matin, peut-être qu'il allume la télévision ou qu'il utilise d'autres appareil qui consomme de l'énergie le matin.

Nous pouvons ainsi remarquer que l'ensemble des ménages consomme beaucoup plus l'énergie électrique le week-end surtout le samedi ; cela peut-être expliquer par le fait qu'ils passent souvent plus de temps chez eux le week-end.

En général, les fonctions graphiques précédentes seront très utiles pour visualiser les différences entre les différents groupes obtenus par notre clustering. Nous en affichons une dernière qui permet de visualiser la consommation moyenne par jour.

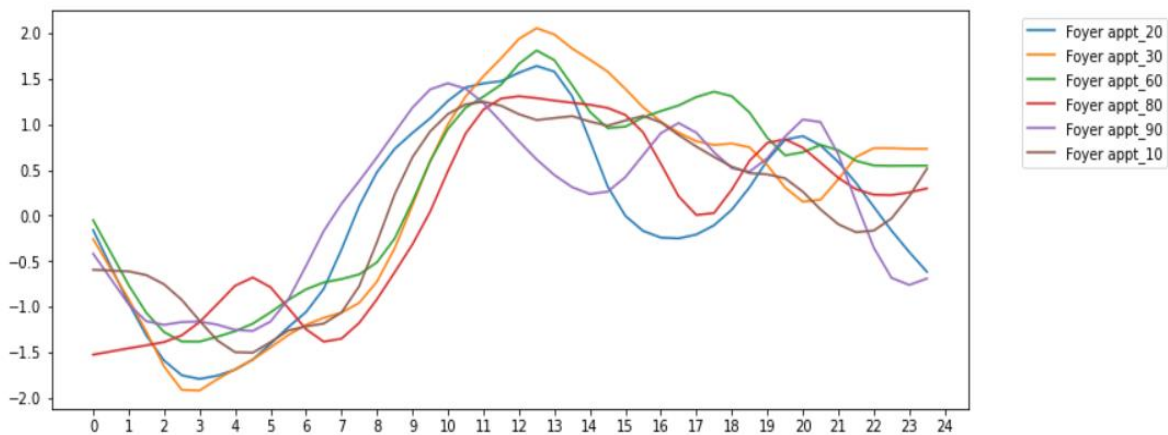


Figure 2 : consommation d'électricité moyenne par jour

L'exemple ci-dessus nous montre que probablement nos données ne présentent pas une variation forte de la consommation brute entre les individus : les courbes suggèrent qu'il n'y en a pas un dépassement léger entre la consommation par un ménage d'un autre. Les différences de comportement entre les ménages sont beaucoup moins évidentes visuellement parlant. Mais cela n'est déduit qu'en se basant sur un ensemble réduit de ménages, rien ne peut nous assurer que c'est le cas pour toutes les données.

Donc afin d'éviter le fait que notre algorithme de clustering aussi identifie la consommation des ménages qui sont beaucoup plus élevée que les autres, en faisant un groupe avec les ménages à forte consommation et un autre avec les ménages à faible consommation, et afin de ne se focaliser que sur les différences de comportement, il pourrait être intéressant de normaliser chaque courbe afin que les différences de consommation brutes ne soient pas identifiées par l'algorithme de clustering.

III. Réduction de la dimension & Extraction des caractéristiques

Cette partie est consacrée aux techniques permettant la réduction de dimension et l'extraction des caractéristiques.

3.1 Principal Components Analysis (PCA)

L'objectif de l'analyse en composantes principales est de représenter un nuage de points par sa projection sur le sous-espace affine de dimension k minimisant l'inertie. Ici, nous avons près de 4368 données différentes pour chacun des foyers, et nous aimerions réduire à un nombre de caractéristiques suffisant pour expliquer 90% de la variabilité des données.

Pour que l'ACP fonctionne, nous devons à nouveau normaliser les données, mais cette fois-ci par caractéristiques. En effet, sans cette normalisation, l'ACP ne sélectionnerait que les caractéristiques ayant le plus de variance, car garder ces caractéristiques permet de garder le plus de variance possible. Dans notre cas, cela reviendrait à ne regarder que les moments où il y a le plus de différences brutes de consommation, ce que nous ne voulons pas, car nous voulons identifier et cibler les différences de comportement à tout moment.

Ceci permet que certaines dates ne prennent pas plus d'importance que d'autres, et donc que l'ACP permette de faire une réelle réduction de dimension.


	0	1	2	3	4	5	6	7	8	9	...	
0	-3.135196	-0.164678	-0.364578	0.637250	-1.517390	2.734195	0.091045	2.571666	3.445291	-0.021527	...	-0.911
1	0.361574	-1.540512	-0.805996	1.019425	1.393532	1.210297	-1.267075	-2.057341	2.070642	-0.641823	...	-1.561
2	1.352010	-2.113286	5.751091	0.326116	-0.630978	-0.315415	0.117600	1.873658	0.147565	1.508805	...	0.631
3	0.509122	-0.888116	-0.716910	-1.947217	2.980348	2.435678	0.473574	0.676666	0.498984	1.324784	...	-0.321
4	0.799264	-0.778398	0.733101	2.200302	-0.660283	0.764601	-0.767409	0.775638	0.155449	-2.041821	...	-0.531
...
95	0.967471	-3.994461	-0.957901	-0.483841	0.594827	-0.931804	0.963131	0.928729	-3.972046	-0.700392	...	-0.411
96	3.508073	-2.531584	-1.602166	-0.921093	-3.134862	0.460745	1.980639	3.459347	-1.165066	-0.128125	...	-0.991
97	-2.319496	-0.514849	0.311074	1.813810	0.358833	-1.734074	0.494962	0.969565	-1.562841	-0.599303	...	-0.111
98	-1.348713	1.518560	0.402513	-2.150179	-0.025476	-2.258834	0.329800	1.732987	2.340974	-1.512598	...	-0.011
99	4.235636	0.834128	0.840164	-1.182023	0.241583	1.301904	1.757075	-2.760346	1.971462	1.153527	...	0.411

100 rows × 58 columns

Figure 3: Les données en PCA

3.2 Réduction de la dimension basée sur les caractéristiques des séries temporelles

Dans cette partie, nous prenons tout le contraire de l'ACP en créant cette fois des variables interprétables pour chaque individu, comme la moyenne de sa consommation, le minimum ou le maximum de la consommation d'un individu.



	Moyenne	Minimum	Maximum	mois_conso_max	mois_conso_min	jour_conso_max	jour_conso_min
appt_1	-0.002530	-2.712846	2.007661	1	3	48	68
appt_2	0.004413	-2.447783	1.964942	2	1	64	7
appt_3	0.015500	-2.551442	1.927615	1	2	69	31
appt_4	0.006204	-2.653337	2.054251	3	1	59	16
appt_5	-0.015020	-2.467466	2.030829	3	1	86	19
...
appt_96	0.012919	-2.379652	2.264872	3	4	22	49
appt_97	-0.003102	-2.649557	2.107669	3	1	90	37
appt_98	-0.006116	-2.731046	2.094527	1	3	20	59
appt_99	0.005128	-2.532689	1.971988	3	2	89	68
appt_100	0.005883	-2.876890	1.961424	3	1	28	24

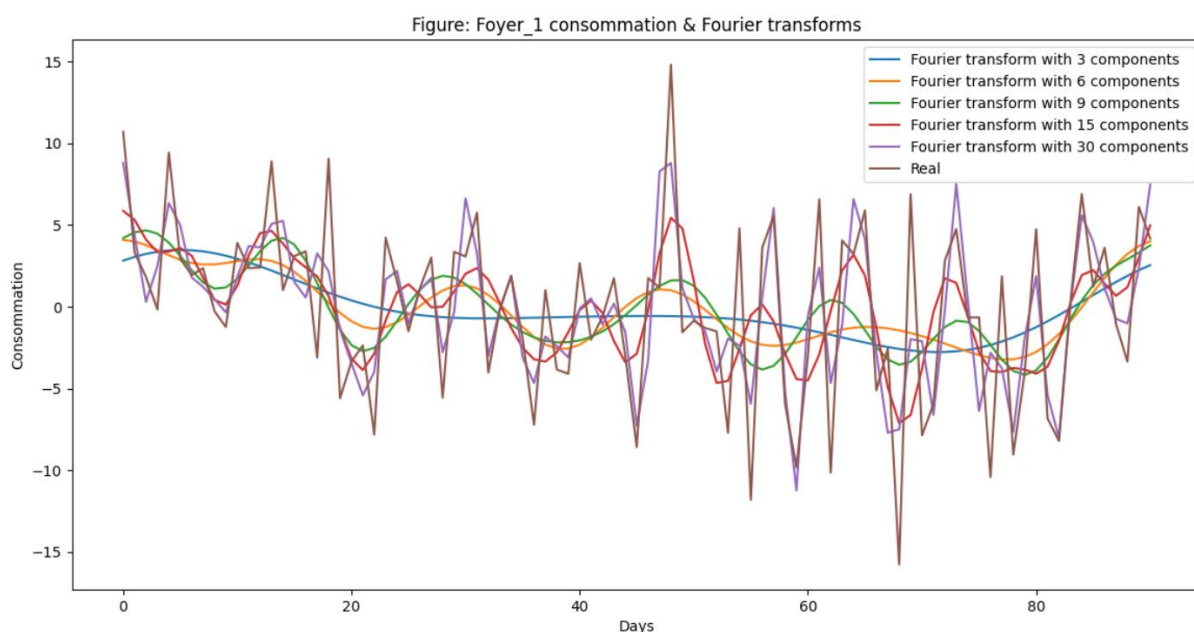
100 rows × 7 columns

Figure 4: Data after applying Dimension reduction based on time series features

Après la création des caractéristiques, il a été important de les normaliser : la plupart des algorithmes de clustering se basent sur la norme euclidienne pour créer des clusters, donc toutes nos caractéristiques doivent avoir le même ordre de grandeur afin que certaines d'entre elles ne prennent pas plus de place que d'autres dans le clustering

3.3 Transformation de Fourier

La transformation de Fourier utilise de nombreuses composantes spectrales pour essayer de former les données. Du domaine temporel, elles sont converties en domaine fréquentiel, puis calculées. Après cela, elle est reconvertie dans le domaine temporel où elle est tracée. La transformation de Fourier en tant qu'indicateur permet d'extraire le cycle prédominant d'une série de données. Dans la figure ci-dessous, nous pouvons visualiser que la composante spectrale de 30 est la plus proche de la consommation réelle : c'est ainsi que nous avons considéré d'avancer avec 30.



IV. Clustering

4.1 Self-Organizing Maps (SOM)

Les cartes auto-organisatrices sont un type de réseau neuronal qui est formé en utilisant l'apprentissage non supervisé pour produire une représentation à faible dimension de l'espace d'entrée des échantillons de formation, appelée carte.

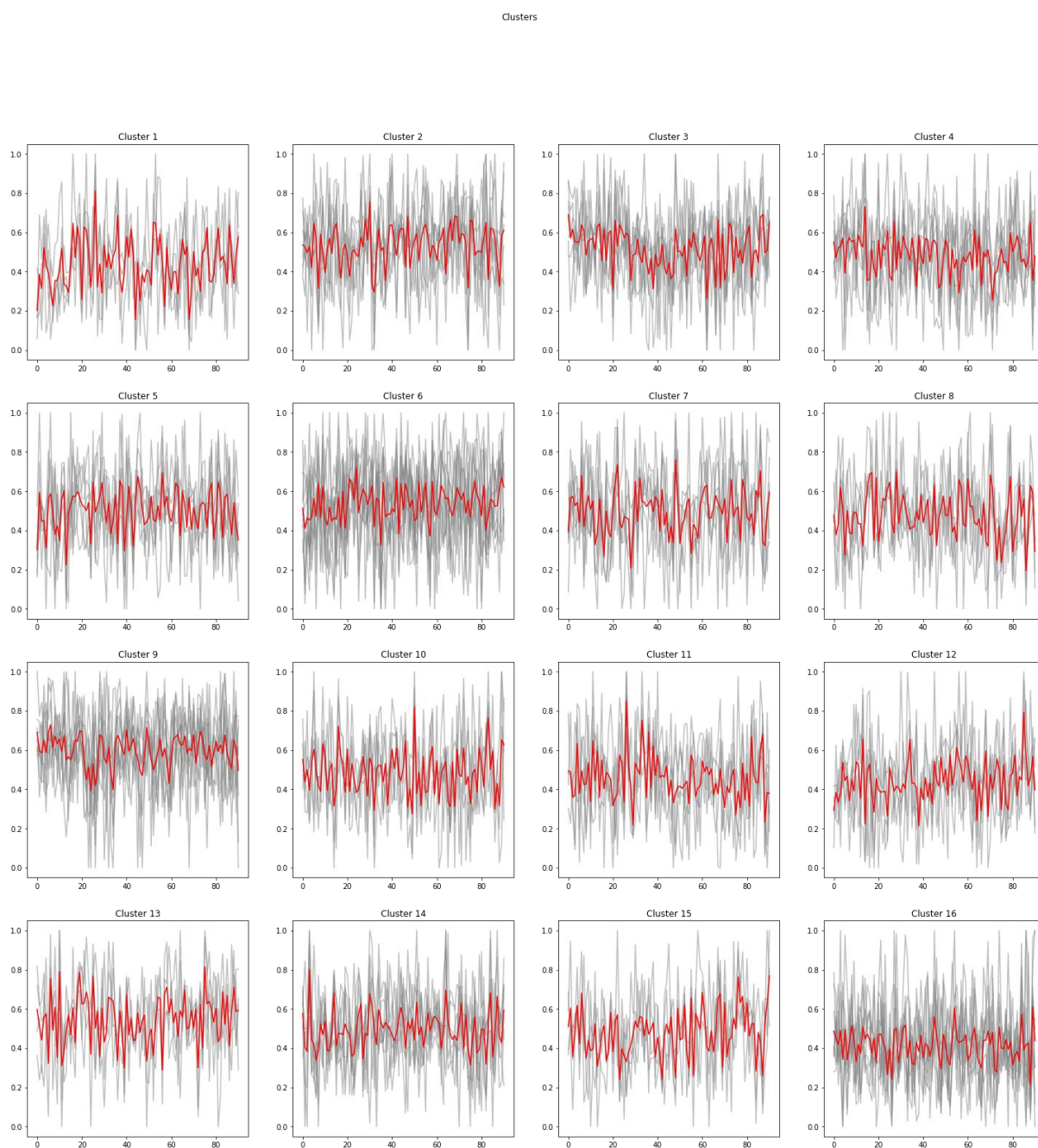
En raison de sa capacité à produire une carte, le SOM est considéré comme une méthode de réduction de la dimensionnalité. Mais dans notre cas, lorsque chaque nœud du SOM est accepté comme médioïdes du cluster, nous pouvons l'utiliser pour le clustering. Pour ce faire, il a fallu supprimer les indices de temps de nos séries temporelles, et au lieu des valeurs mesurées de chaque date, nous devrions les accepter comme différentes caractéristiques et dimensions d'un seul point de données.

Pour l'implémentation de l'algorithme SOM, les paramètres suivants ont été définis :

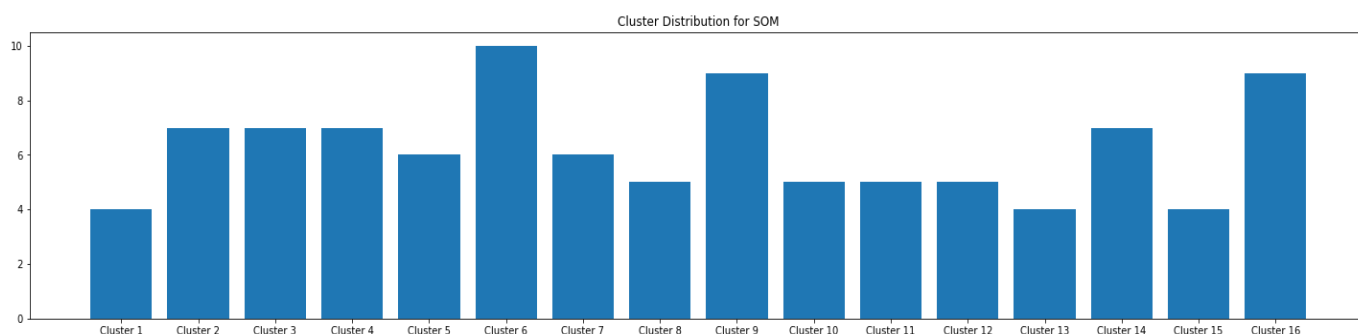
Tableau 1: Configuration de SOM

Paramètres	Valeur
Sigma	0.3
Taux d'apprentissage	0.5
Nombre d'itérations	50000
Taille de la carte	Racine carrée du nombre de séries
Poids	Initialisation aléatoire

Après l'entraînement, nous avons tracé résultats : pour chaque cluster, nous avons tracé toutes les séries, un peu en transparence et en gris, et afin de voir le mouvement ou la forme du cluster, nous avons pris la moyenne du cluster et puis nous avons tracé la série moyenne obtenue en rouge (Figure ci-dessous). Comme nous pouvons le visualiser sur le graphique ci-dessous, SOM a parfaitement regroupé les 100 séries différentes en 16 clusters.



Nous pouvons visualiser la distribution de nos séries chronologique en grappes, comme c'est démontré dans la figure ci-dessous.



Nous avons bien regroupé nos séries, mais comment savoir quelle série appartient à quel groupe ? N'est-ce pas là tout le but du clustering ?

Comme nous pouvons le voir, chaque nœud (ou plusieurs nœuds dans certains cas) représente un cluster. Par conséquent, nous pouvons trouver quelle série appartient à quel groupe en vérifiant le nœud gagnant de chaque série.

Series	Cluster
appt_37	Cluster 1
appt_58	Cluster 1
appt_96	Cluster 1
appt_71	Cluster 1
appt_30	Cluster 10
appt_99	Cluster 10
appt_10	Cluster 10
appt_78	Cluster 10
appt_16	Cluster 10
appt_63	Cluster 11
appt_32	Cluster 11
appt_77	Cluster 11
appt_13	Cluster 11
appt_84	Cluster 11
appt_74	Cluster 12
appt_54	Cluster 12
appt_92	Cluster 12

Figure 5: Mappage des clusters

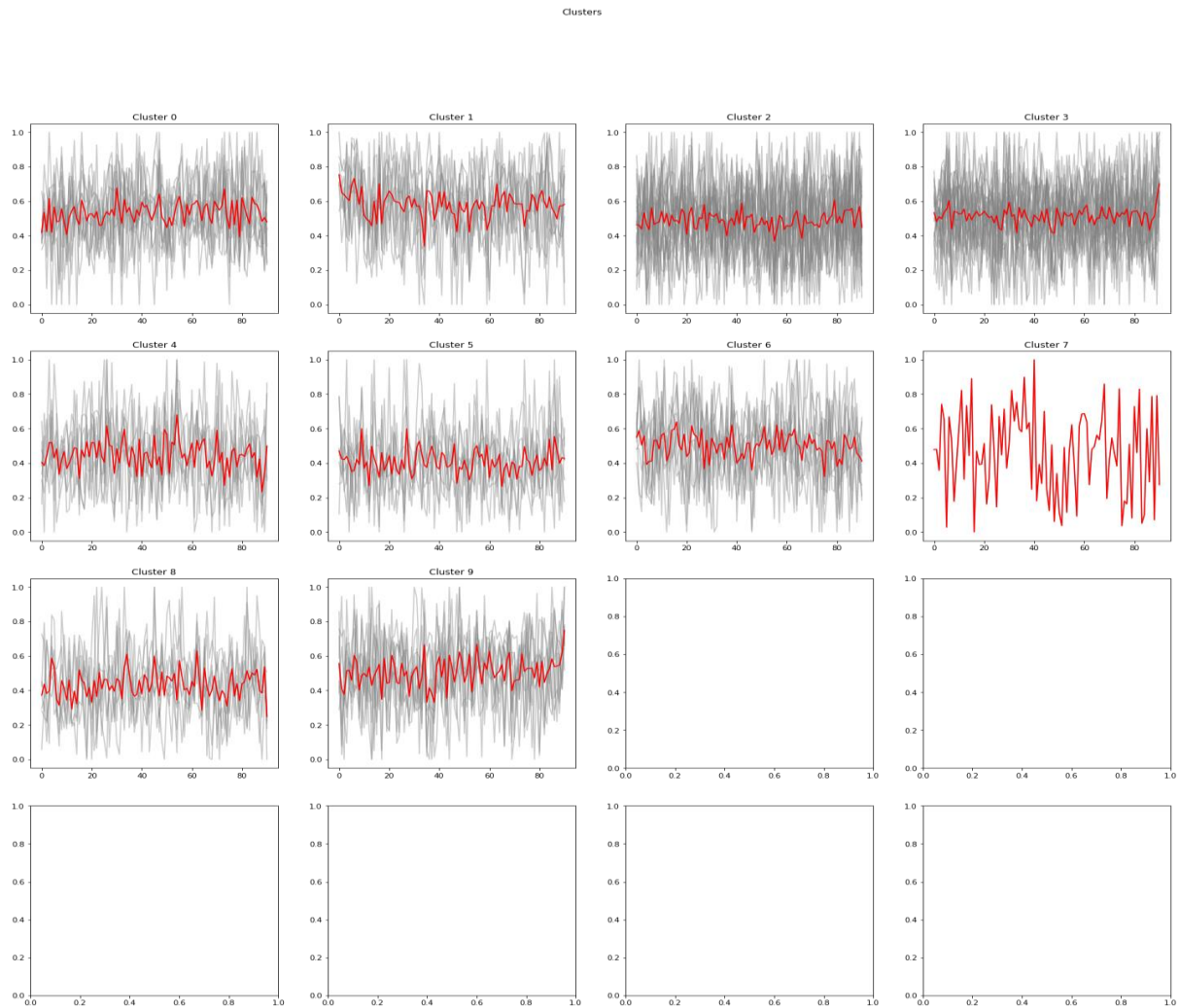
4.2 K-means

Le clustering K-means est une méthode qui vise à regrouper n entrées en k clusters dans lesquels chaque point de données appartient au cluster avec la moyenne la plus proche (centroïde du cluster).

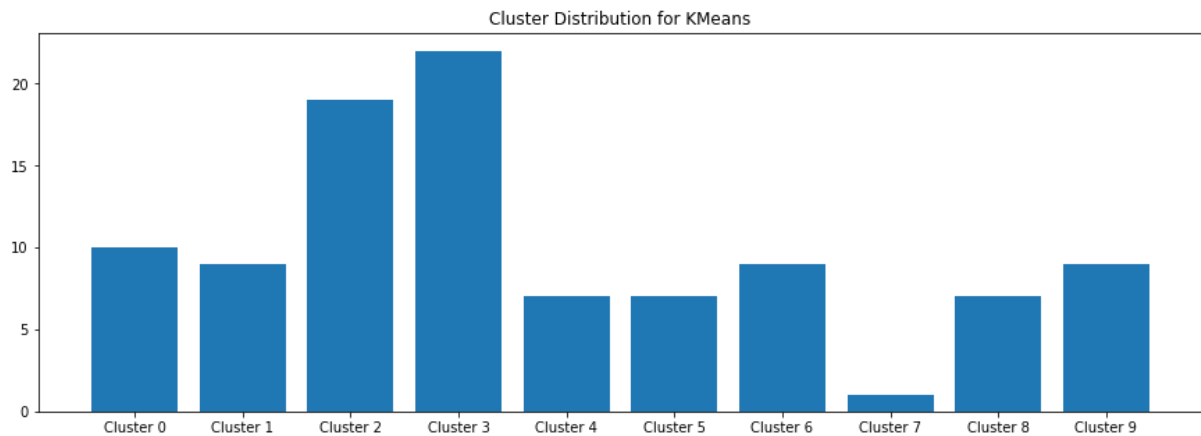
Afin de regrouper nos séries avec k-means, la chose essentielle à faire est, comme nous le faisons avec SOM, de supprimer les indices de temps de nos séries temporelles, et au lieu des valeurs mesurées de chaque date, nous devrions les accepter comme différentes caractéristiques et dimensions d'un seul point de données. Une autre chose importante à faire

est de sélectionner la métrique de distance : nous utiliserons le Dynamic Time Warping (DTW).

Après l'entraînement, on a obtenu le tracé suivant : les k-means ont regroupé les 100 séries différentes en 10 clusters. L'un des clusters ne contient qu'une seule série temporelle qui peut être considérée comme une valeur aberrante. (Figure ci-dessous)



Il semble aussi, d'après la distribution des séries temporelles, que le k-means a regroupé environ 21 des séries temporelles dans le cluster 3 : ce qui est un peu asymétrique.

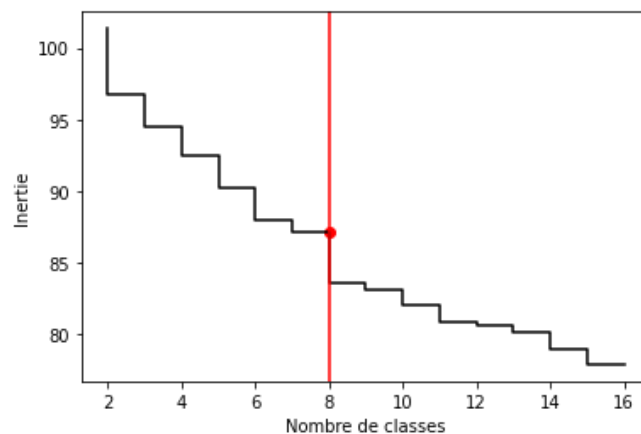


4.3 Classification Hiérarchique (CAH)

Nous passons maintenant un regroupement par classification hiérarchique. L'avantage d'une telle technique est que nous pouvons décider ici de couper le dendrogramme à l'endroit souhaité, pour obtenir des groupes de la taille désirée.

4.3.1 Méthode directe

On visualise sur le graphique ci-dessous un grand saut d'inertie pour 8 classes. Cela suggère bien que si on découpe en 8, ça serait pertinente.



Ce qui a entraîné vers un découpage à la hauteur $t=85$

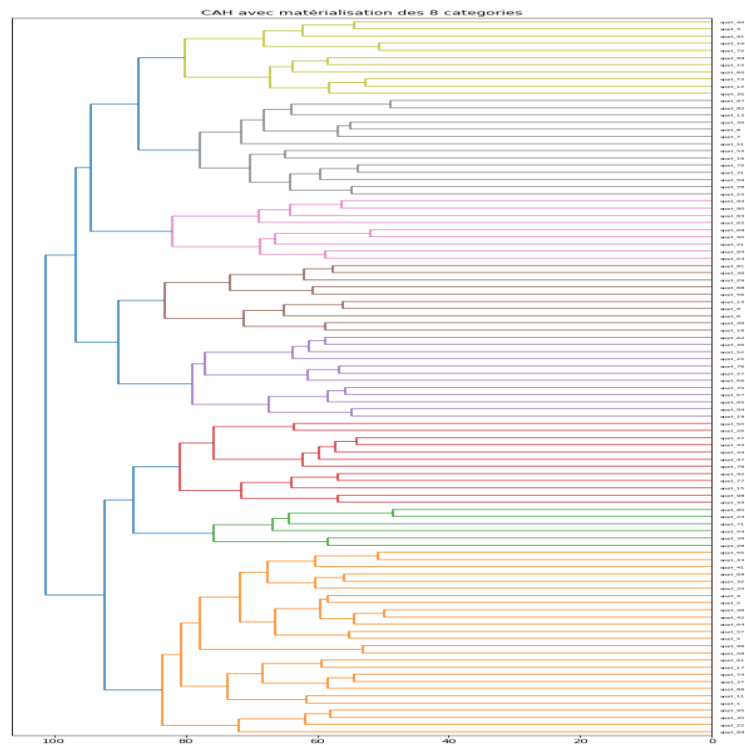
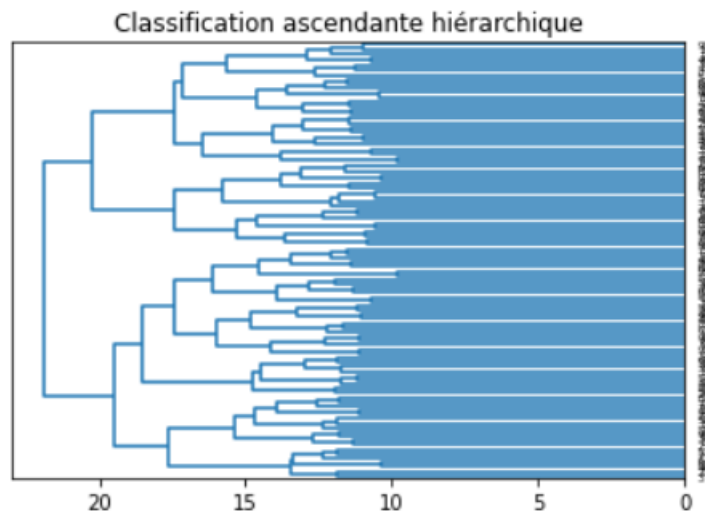


Figure 6: CAH avec dématérialisation de 8 catégories

4.3.2 Méthode basé sur l'extraction des caractéristiques

Afin d'évaluer nos résultats, nous utilisons l'indice Calinski Harabasz : **le rapport entre la variance inter-groupes et la variance intra-groupe**. Malheureusement, le résultat obtenu était faible < 3.9 pour ce type de méthode.

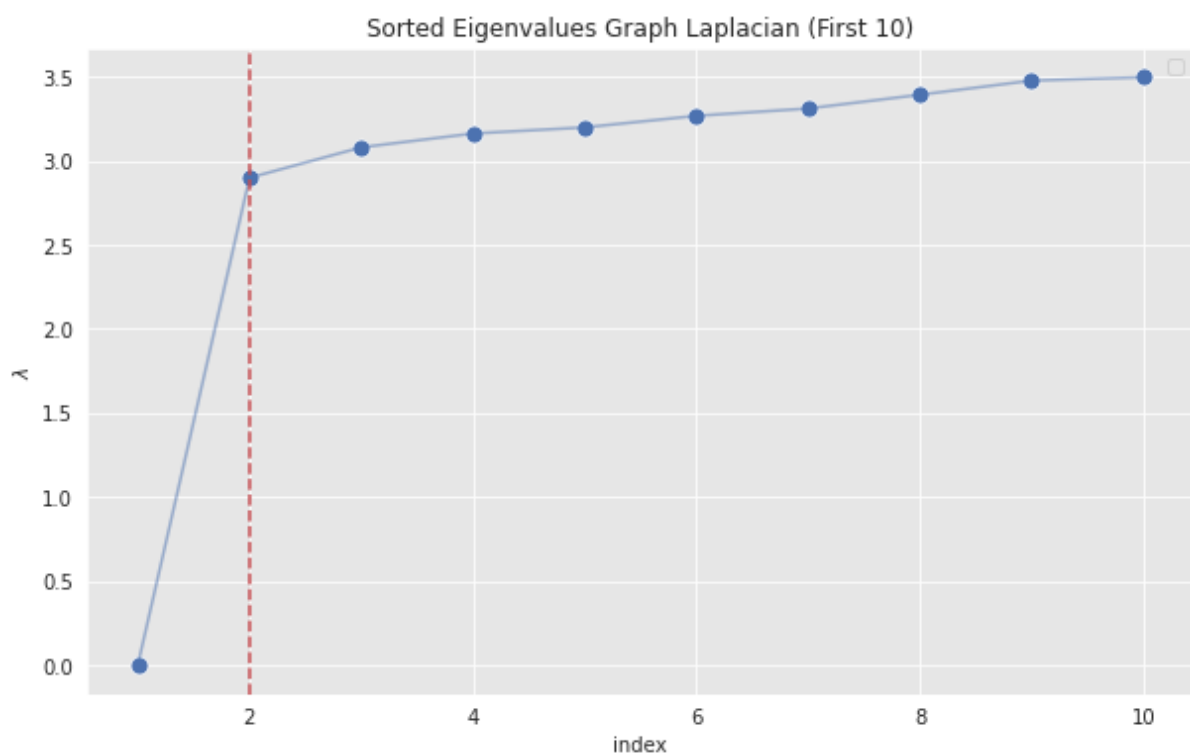


Indice de Calinski Harabasz : 2.7891164020785926

ID
 0
 1 53
 2 47 ACP 2

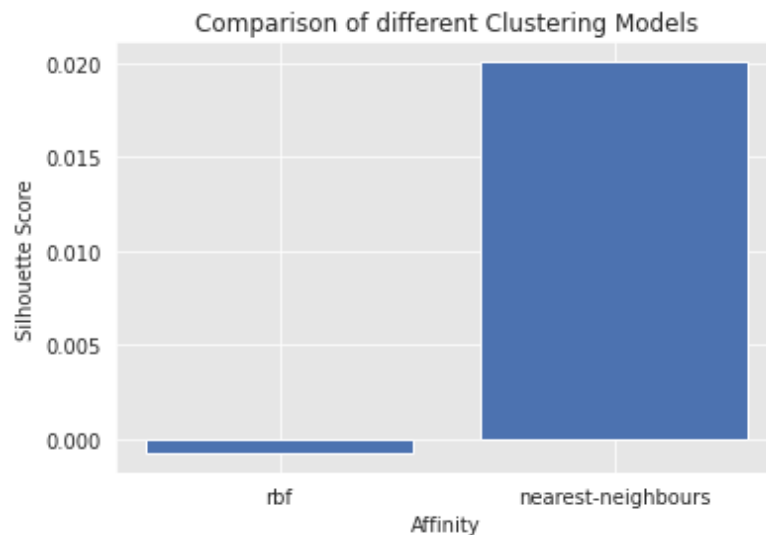
4.4 Classification spectrale

Le clustering spectral est une approche flexible pour trouver des clusters lorsque nos données ne répondent pas aux exigences des autres algorithmes courants. Les valeurs propres du Laplacien des graphes peuvent être utilisées pour trouver le meilleur nombre de clusters, et les vecteurs propres peuvent être utilisés pour trouver les étiquettes des clusters.



En général, nous recherchons souvent le premier grand écart entre les valeurs propres afin de trouver le nombre de clusters exprimés dans nos données.

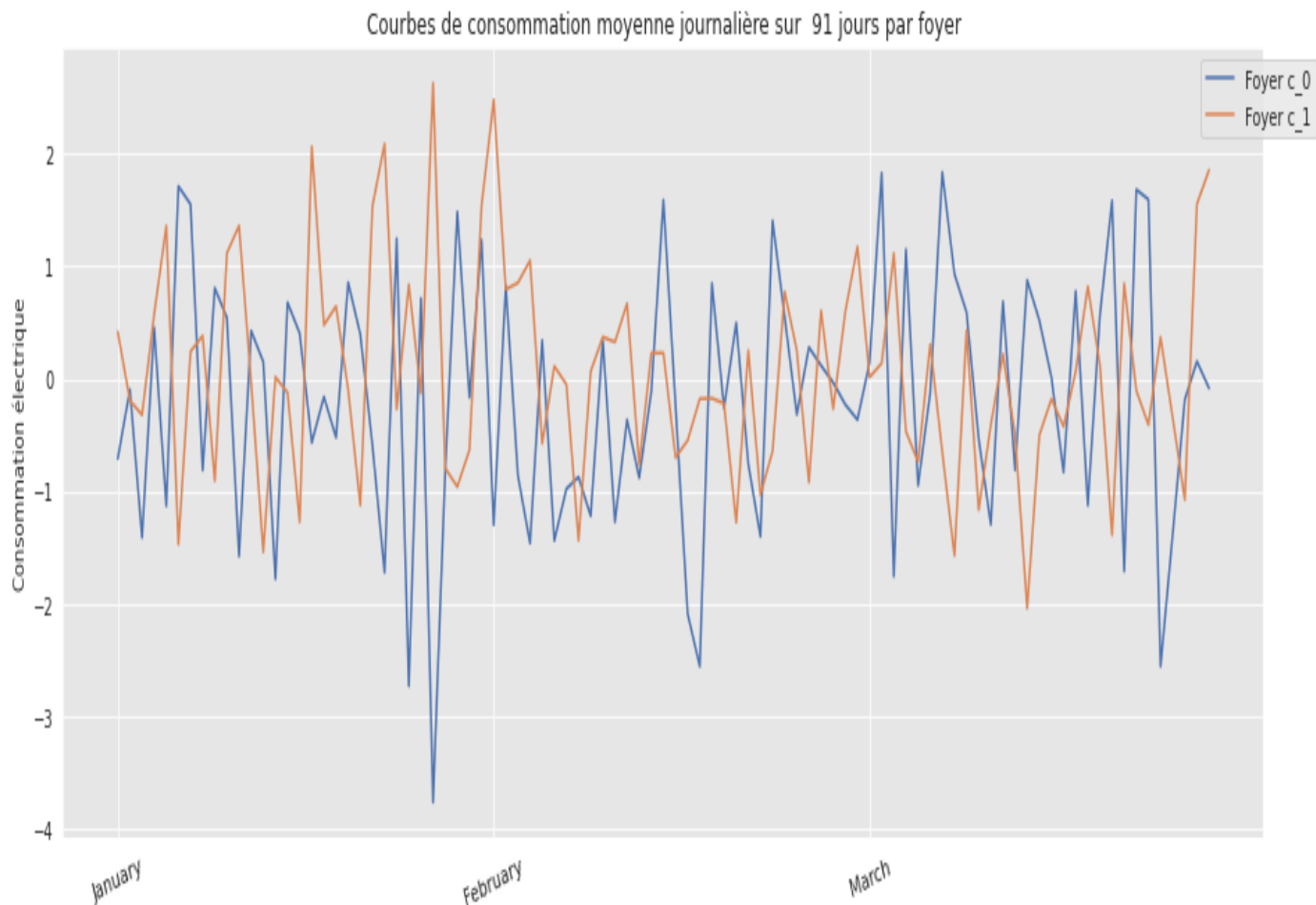
La figure ci-dessous nous permet de choisir l'affinité la plus convenable pour le clustering finale, et nous pouvons visualiser que 'nearest_neighbors' est meilleur que 'rbf' : donc l'affinité = 'nearest_neighbors'



Après l'application de la classification spectrale sur nos données, chaque ménage a été attribué à un cluster ; c_0 représente le 1^{er} cluster et c_1 représente le 2^{ème} cluster. (figure ci-dessous)

```
data1['cluster']  
  
appt_1    c_1  
appt_2    c_1  
appt_3    c_1  
appt_4    c_1  
appt_5    c_0  
...  
appt_96   c_1  
appt_97   c_1  
appt_98   c_1  
appt_99   c_0  
appt_100  c_0  
Name: cluster, Length: 100, dtype: object
```

Maintenant, nous pourrions réaliser des visualisations communes à l'ensemble des séries temporelles, puisqu'ils sont regroupés en clusters.



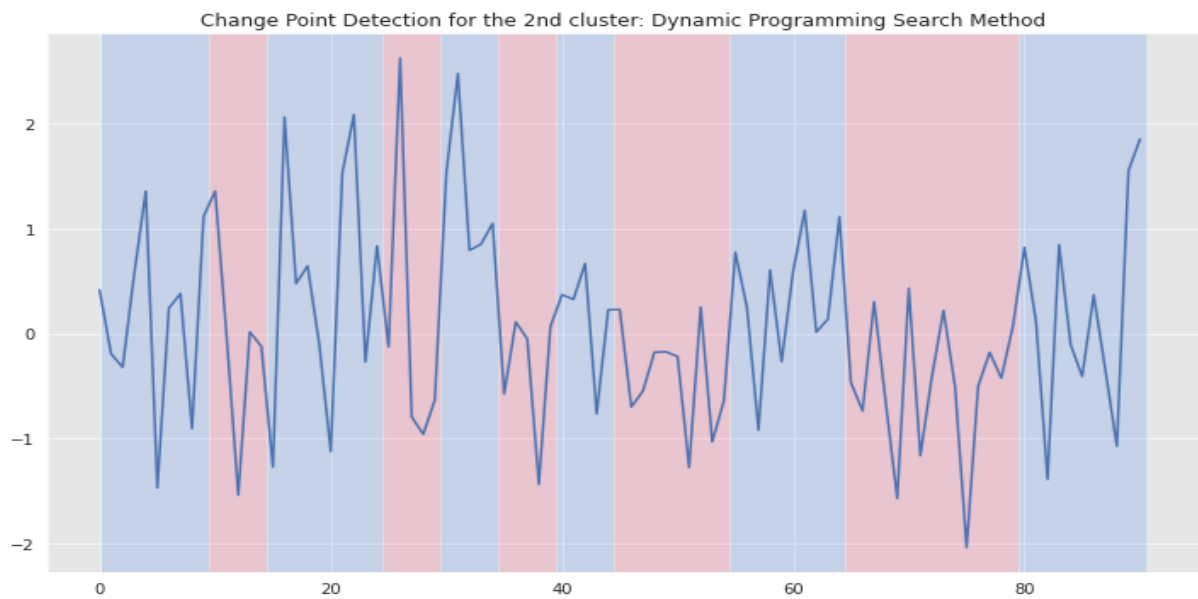
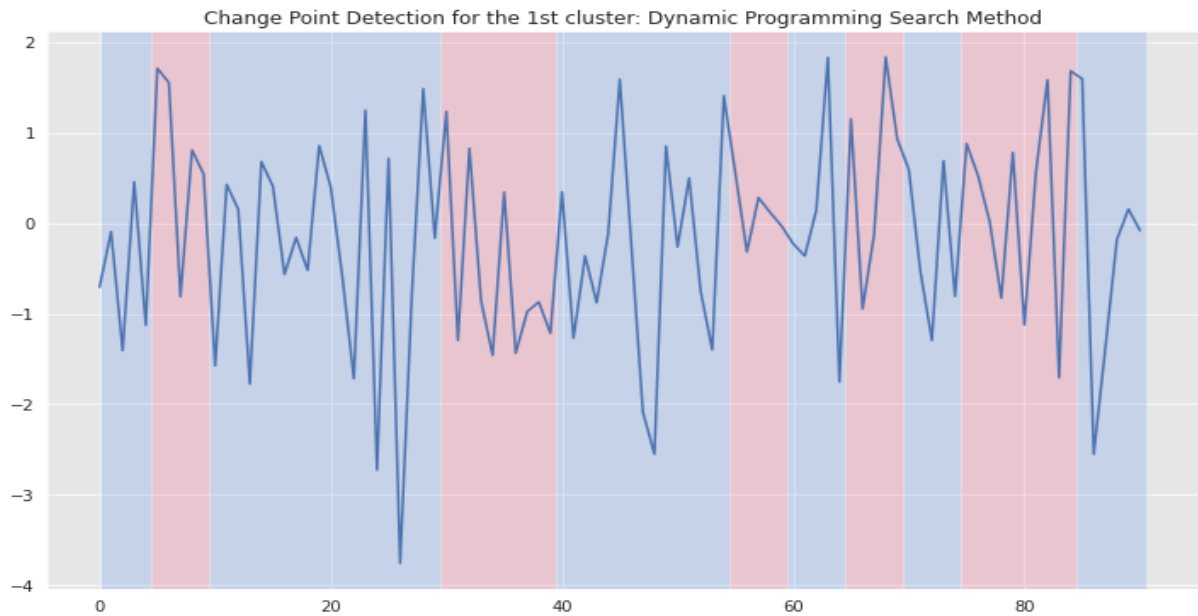
On note que les 2 clusters (c_0 et c_1) semblent être différents : nombreux sont les jours où lorsque la consommation du groupe c_1 devient élevée, la consommation de la part du groupe c_0 devient faible ; ceci peut être expliqué par le fait que les activités exercées par les ménages du 1^{er} groupe sont différentes par rapport aux ménages du 2^{ème} groupe, ou le nombre d'individus constituant chacun des foyers d'un groupe se différencie de l'autre groupe.

V Ruptures

Dans cette partie, nous allons prendre en main les clusters obtenus précédemment et puis on crée un signal bruyant constant par morceaux, effectue une détection des points de changement par noyau pénalisé et affiche les résultats (des couleurs alternées marquent les régimes réels et des lignes pointillées marquent les points de changement estimés).

5.1 Algorithme de programmation dynamique

Il s'agit d'une méthode exacte, qui a un coût de calcul considérable de $O(Qn^2)$, où Q est le nombre maximal de points de changement et n est le nombre de points de données.

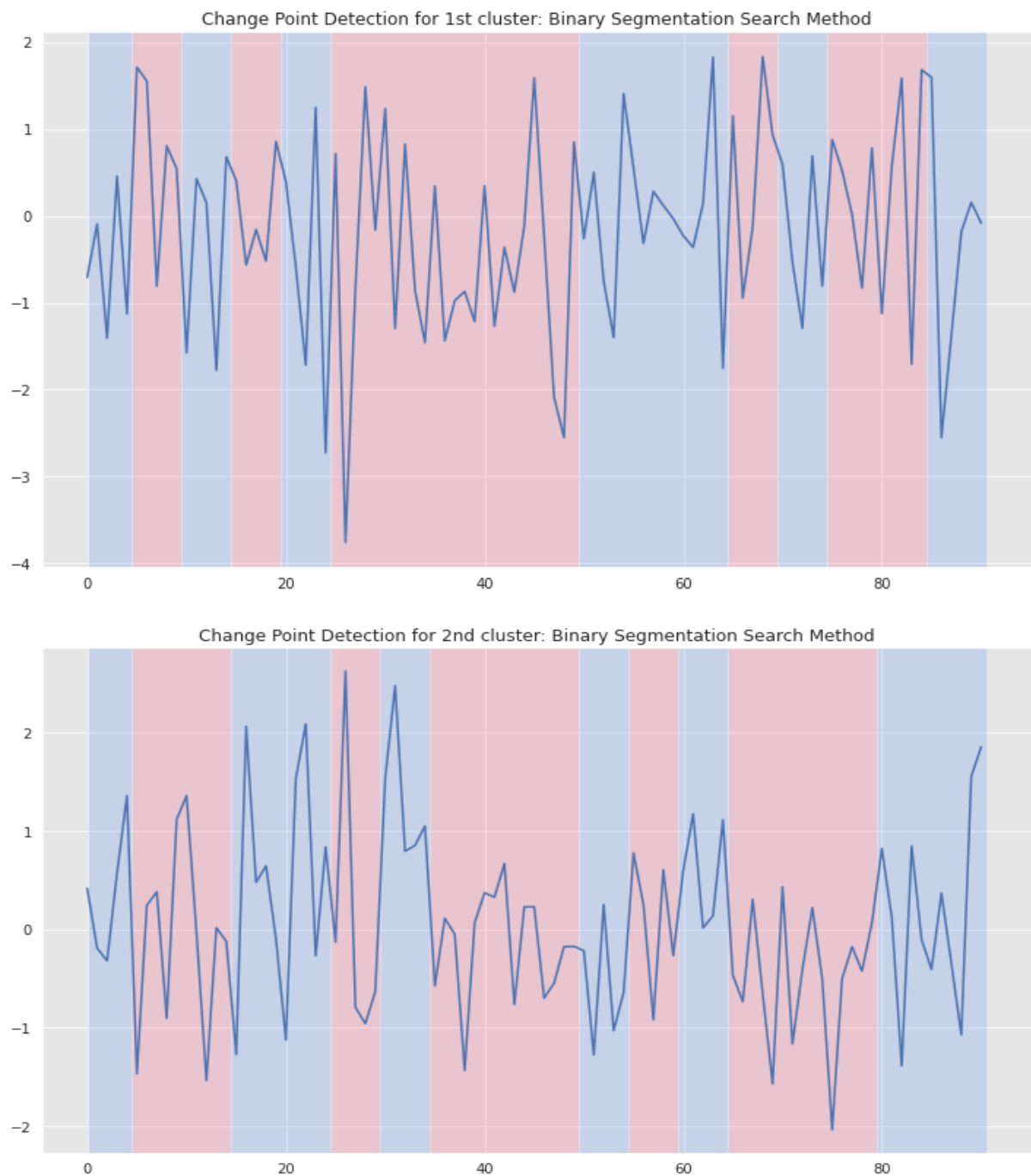


5.2 Binary Segmentation Search Methods

Cette méthode est sans doute la plus établie dans la littérature. La segmentation binaire est une méthode approximative avec un coût de calcul efficace de $O(n \log n)$, où n est le nombre de points de données.

L'algorithme fonctionne en appliquant de manière itérative une méthode de point de changement unique à l'ensemble de la séquence pour déterminer si une division existe. Si une

division est détectée, la séquence se divise en deux sous-séquences. Le même processus est alors appliqué aux deux sous-séquences, et ainsi de suite.



Comme nous pouvons le voir dans les graphiques ci-dessus, les points de changement détectés dans la séquence diffèrent en fonction de la méthode de recherche utilisée

VI Conclusion

Dans ce projet, nous avons eu l'opportunité d'exploiter de différentes méthodes de la classification non supervisé sur des données temporelles.

Nous avons tout d'abord réalisé une analyse descriptive de nos séries temporelles, tout en les visualisant dans une échelle journalière. Ensuite nous avons essayé de les manipuler par les méthodes de réduction de la dimension et l'extraction des caractéristiques. Cela étant, nous avons appliqué un ensemble de méthodes de clustering afin de regrouper les séries en des clusters homogènes tout en observant les comportements des groupes obtenus. Les séries temporelles linéaires étant des données relativement complexes à manipuler, il faut toujours donc appliquer les transformations (comme les normalisations) de manière appliquée et réfléchie afin de conserver de bons résultats. Il est également important de définir ce que l'on recherche.

Lien du notebook associé :

<https://colab.research.google.com/drive/1MKYGGNqc1fmYowifWWV0Bt2-TqJrCcz1?usp=sharing>