UNIVERSITÉ DE PARIS

Master 2 - Apprentissage Mchine pour la Science des Données

Rapport de projet - Apprentissage non supervisé

Human activity recognition with smartphones

Travail réalisé par:
Assia BOURAÏ
Félix DORNIER
Soufiane SEBBATA

Chargé de cours:
Allou SAME

Décembre 2021

Contents

1 Introduction

Le présent document constitue une synthèse du travail réalisé dans le cadre du projet de l'UE Apprentissage Non Supervisé au sein de l'Université de Paris.
L'objectif de ce projet est d'utiliser la classification non supervisée en appliquant les différentes méthodes vues en cours pour aider à reconnaître, à partir de données issues de smartphones, différentes classes d'activités physiques de personnes : (1) marcher, (2) monter les escaliers, (3) descendre les escaliers, (4) s'asseoir, (5) se lever, (6) s'allonger.

2 Description des données

Durant la collecte de données, 9 variables ont été enregistrées toutes les 0.02 seconde : 3 accélérations réelles (suivant les axes x, y, z), 3 accélérations estimées (suivant les axes x, y, z) et 3 vitesses (suivant les axes x, y, z) puis découpées en fenêtres temporelles glissantes de 128 observations (chaque fenêtre correspond à un extrait de 2.5 secondes). Ces données ont été rangées dans 9 tableaux de taille identique (347 $\times$ 128) dont chaque ligne correspond à une même fenêtre temporelle de 128 observations.
On peut donc dire que l'on a 9 matrices, et chaque matrice contient 347 lignes (nombre d'observations) et 128 colonnes (points de temps). C'est donc assimilable à des séries temporelles que l'on veut classifier suivant les similarités entre elles.
On charge d'abord nos différents tableaux pour pouvoir travailler dessus sur le notebook R. On obtient des tableaux qui ressemblent à ce qui suit (voir figure 1) :

V1 <dbl>	V2 <dbl>	V3 <dbl>	V4 <dbl>	V5 <dbl>	V6 <dbl>	V7 <dbl>	V8 <dbl>
1.01281700	1.02283300	1.02202800	1.01787700	1.02368000	1.01697400	1.01774600	1.01926300
1.01885100	1.02238000	1.02078100	1.02021800	1.02134400	1.02052200	1.01979000	1.01921600
1.02312700	1.02188200	1.01917800	1.01586100	1.01289300	1.01645100	1.02033100	1.02026600
1.01768200	1.01814900	1.01985400	1.01988000	1.01912100	1.02047900	1.02059500	1.01634000
1.01995200	1.01961600	1.02093300	1.02306100	1.02224200	1.02086700	1.02193900	1.02230000
1.02127800	1.01887800	1.01921900	1.02198600	1.02224600	1.02124000	1.01960900	1.01894700
1.01645800	1.01608400	1.01884700	1.02397300	1.01974300	1.01642700	1.02191900	1.02507700
1.02399600	1.02448500	1.02512000	1.02388400	1.02124400	1.01987600	1.01809500	1.01606800
1.02673600	1.02317600	1.02186000	1.02036700	1.02137300	1.02330500	1.02169800	1.02215300
1.01928900	1.01923300	1.01982200	1.01929900	1.02047500	1.02099600	1.01984600	1.01936700

1-10 of 347 rows | 1-8 of 128 columns Previous 1 2 3 4 5 6 ... 35 Next

Figure 1: Aperçu du tableau accm_x

Nous avons donc au total 1152 colonnes sur 347 lignes, nous allons combiner les différents tableaux dans un même dataset pour pouvoir effectuer une réduction de la dimension et observer les données sur l'espace réduit. On modifie le nom des colonnes pour ne pas avoir de problème de noms répétés.

Analyse de la distribution des classes

D'après la figure 2 ci dessous, nous pouvons voir que le nombre d'observations pour les classes "allonger", "se lever", "monter","descendre" et "assis" est distribué de manière presque égale ce qui n'est pas le cas pour la classe "marcher" qui domine les autres classes.

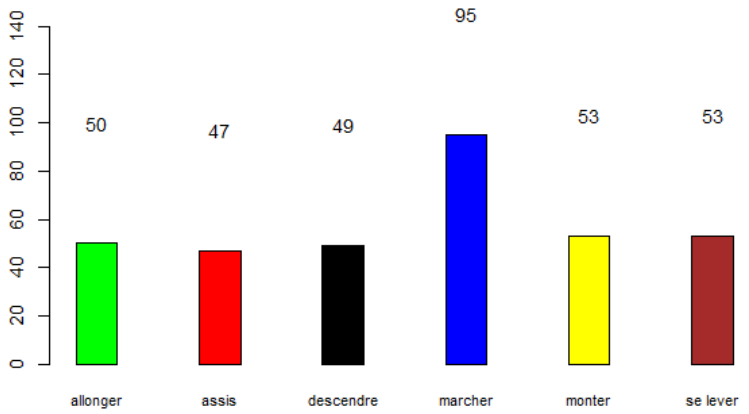

Figure 2: Analyse de la distribution des classes

3 Méthodes utilisées

3.1 Réduction de la dimension avec ACP

Avant d'entammer la classification des données, une réduction de dimension s'avère essentielle pour notre analyse. L'analyse en composantes principales (ACP) nous permettra ainsi de réduire la dimension des données multivariées à un petit nombre de composantes principales, qui peuvent être visualisées graphiquement, en perdant le moins possible d'information.
On utilise les librairies "FactoMineR" et "factoextra". La fonction prcomp() nous permet d'obtenir l'analyse en composantes principales. Ensuite, on utilise "get_eigenvalue()" pour obtenir les valeurs propres (voir figure 3).

	eigenvalue <dbl>	variance.percent <dbl>	cumulative.variance.percent <dbl>
Dim.1	2.489181e+02	2.160747e+01	21.60747
Dim.2	7.840121e+01	6.805661e+00	28.41313
Dim.3	6.940691e+01	6.024906e+00	34.43804
Dim.4	5.851983e+01	5.079847e+00	39.51788
Dim.5	5.282261e+01	4.585296e+00	44.10318
Dim.6	4.560627e+01	3.958877e+00	48.06206
Dim.7	3.124148e+01	2.711934e+00	50.77399
Dim.8	2.805626e+01	2.435439e+00	53.20943
Dim.9	2.733724e+01	2.373024e+00	55.58246
Dim.10	2.496655e+01	2.167235e+00	57.74969

1-10 of 347 rows Previous 1 2

Figure 3: Valeurs propres des composantes de l'ACP

Nous examinons les valeurs propres pour déterminer le nombre de composantes principales à prendre en considération (voir figure 4).

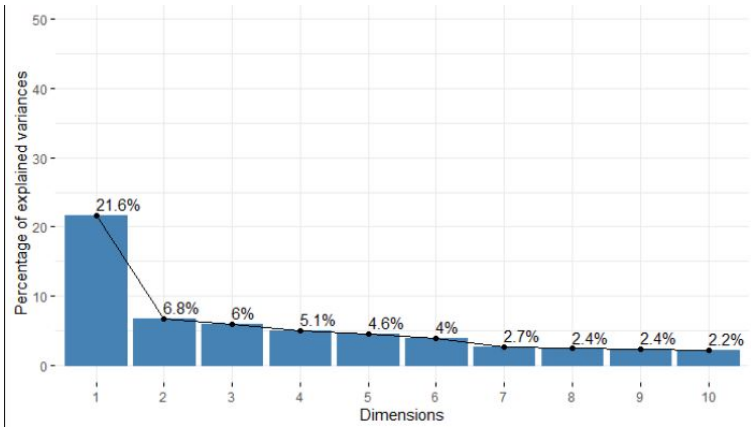

Figure 4: Visualisation des valeurs propres

La figure 4, montre qu'environ 29% de la variance totale est expliquée par les deux premières valeurs propres, et les 58 premières composantes principales expliquent 90% de la variance totale.

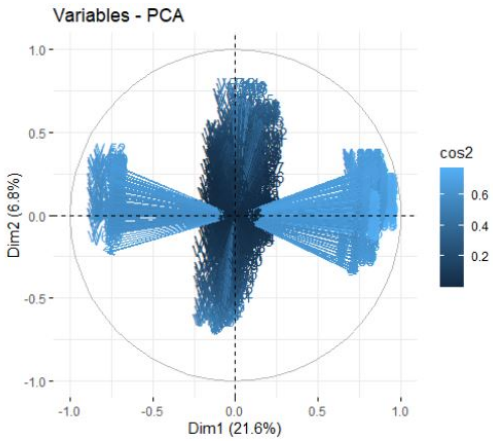

Figure 5: Cercle de corrélation

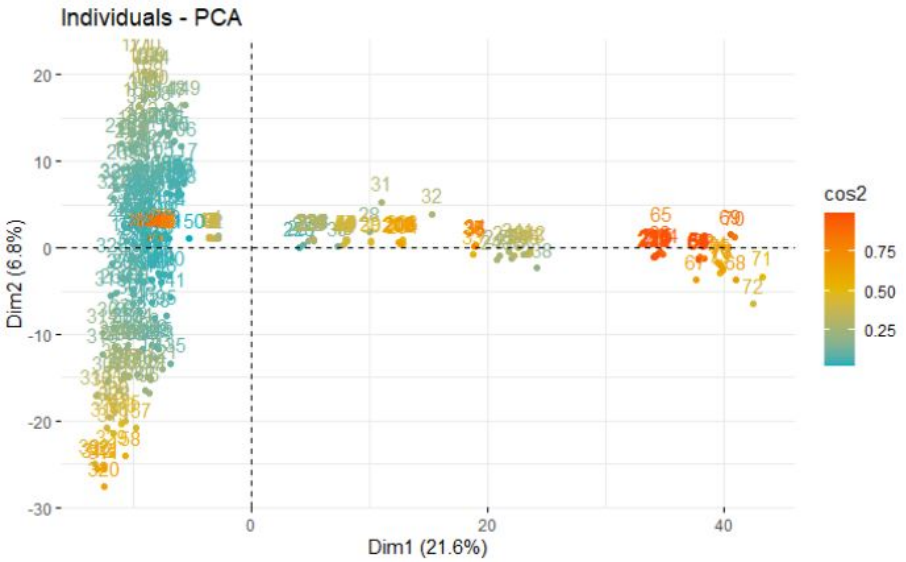

Figure 6: Représentation 2D des données sur l'espace réduit

On remarque que l'espace réduit ne permet pas de distinguer clairement les six clusters. On observe à gauche un grand nuage de points, puis environ trois autres petits clusters distribués sur la longueur.

La figure 7 ci dessous nous montre une autre représentation de l'espace réduit avec l'ACP en mettant en avant les 6 classes.

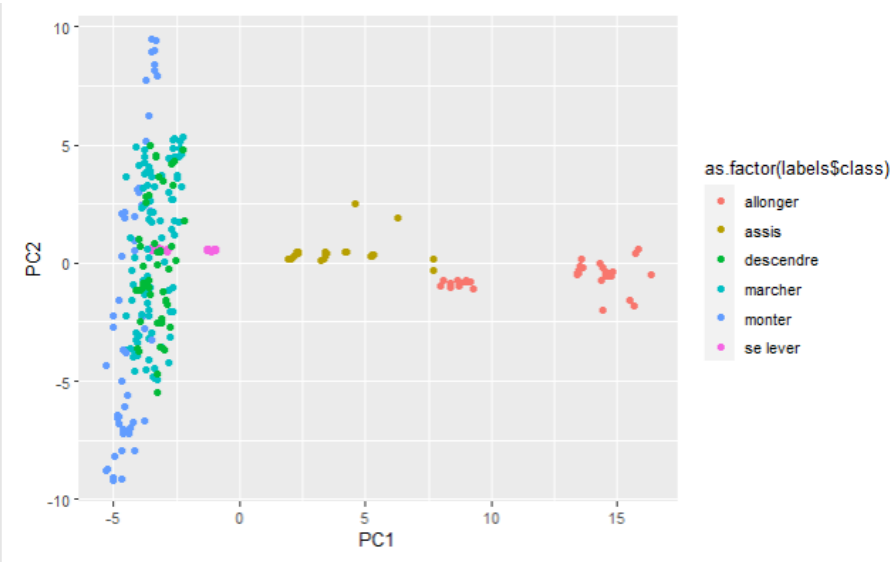

Figure 7: Visualisation 2D de l'ACP

3.1.1 Classification ascendante hiérarchique (CAH)

La première méthode de classification que nous avons appliqué sur les données est la classification hiérarchique avec hclust().

Nous avons appliqué la méthode d'abord sur l'espace d'origine puis sur la réduction PCA en prenant

en considération les 58 premières dimensions. Les figures 8 et 9 ci dessous montrent les dendrogrammes obtenus à l'issue de la classification.

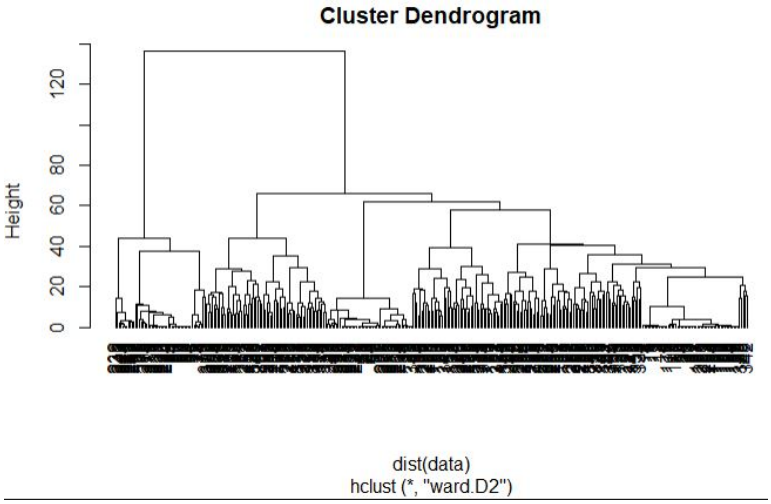

Figure 8: CAH sur l'espace d'origine

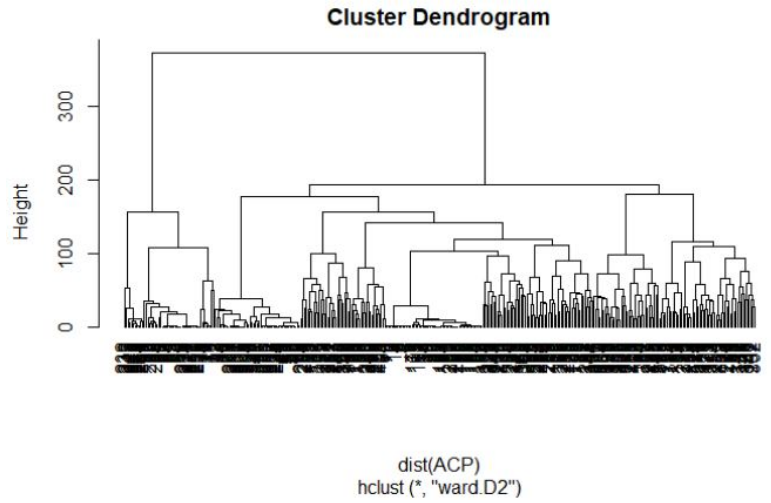

Figure 9: CAH sur l'espace réduit

On remarque sur les deux figures qu'il y a une première séparation en deux clusters. Cela peut pousser à penser que le modèle distingue entre les phases dynamiques et les phases statiques, mais l'une des deux branches contient beaucoup moins d'éléments que l'autre.
Comme on peut le voir sur la figure 10, il y a bien une différence entre les deux figures : on retrouve k optimal égal à 2 pour le premier et k=6 pour le deuxième (PCA).

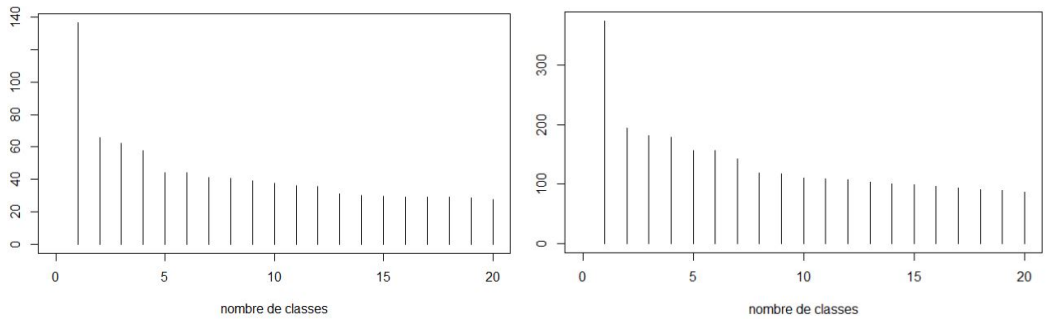

Figure 10: Détermination du nombre de cluster optimal (Données d'origine à gauche, données réduites à droite).

L'avantage avec l'espace réduit c'est que l'on peut visualiser la distribution des clusters sur ce dernier comme le montre la figure 11.

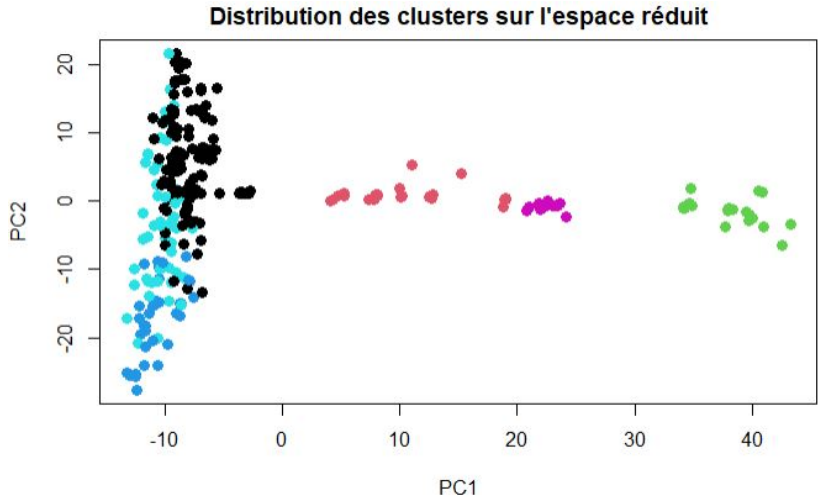

Figure 11: Visualisation des clusters obtenus après le clustering sur les deux composantes principales.

Le clustering appliqué sur PCA atteint un score NMI de 53% contre 58% sur l'espace d'origine.

3.1.2 K-means

On applique maintenant la méthode kmeans sur l'espace d'origine puis sur les espaces réduit.
En premier lieu , on détermine le nombre de clusters optimal que nous donne la méthode. La figure 12 montre le silhouette score en fonction du nombre de clusters.

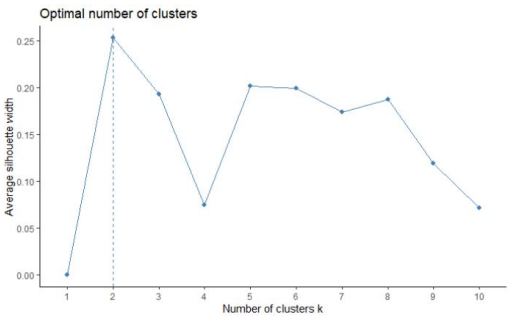

Figure 12: Visualisation du nombre de cluster optimal en focntion du silhouette score.

La méthode des silhouettes affiche la meilleure valeur pour k=2 (deux clusters). Il semblerait qu'ici aussi le modèle distingue entre les deux phases dynamiques et statiques.
On essaiera quand même de faire un clustering pour distinguer les six activités séparément.

Le clustering K-means sur les données d'origine donne un score de NMI égal à 52

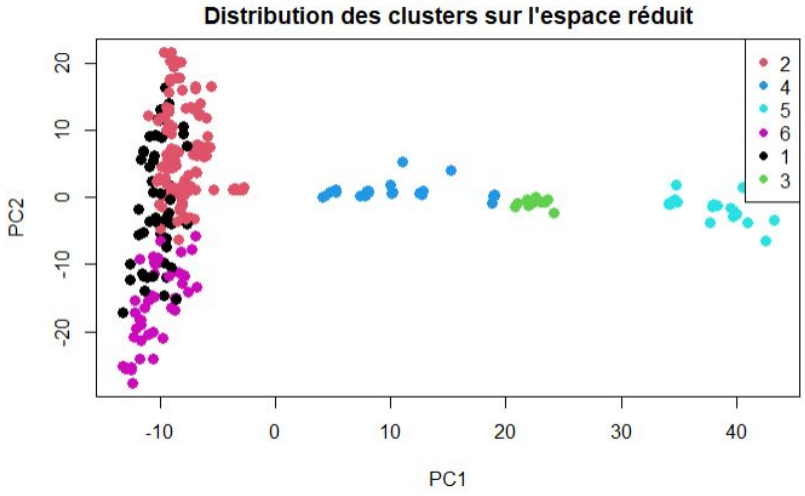

Figure 13: Visualisation des clusters obtenus après le K-means sur l'ACP

On obtient un score d'NMI de 53% sur l'espace réduit avec PCA. On peut donc dire le K-means sur l'espace réduit est légèrement meilleur en terme de NMI mais il a quand même du mal à bien séparer les clusters, ceci est dû à la qualité de l'espace réduit.

3.1.3 Clustering Spectral

Une troisième méthode que l'on a appliquer sur l'espace réduit avec l'ACP est le clustering spectral.

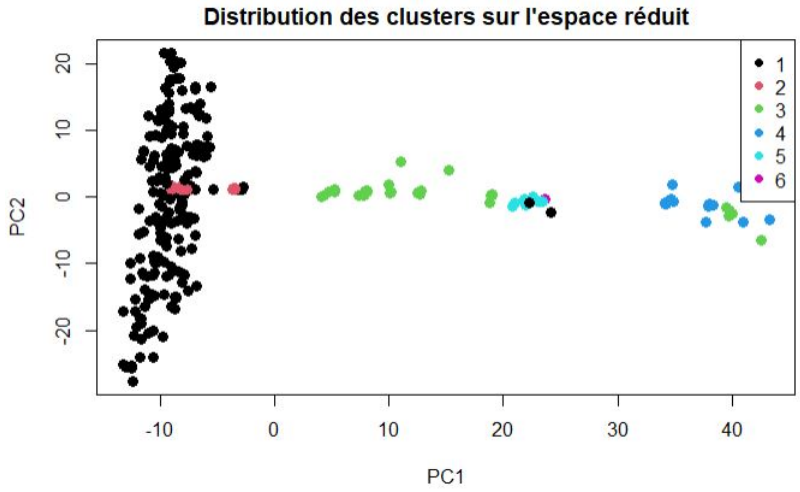

Figure 14: Visualisation des clusters obtenus avec le clustering spectral sur les composantes principales de l'ACP.

On obtient un score NMI de 55% sur l'espace réduit. Par contre, la figure ci-dessus démontre d'un mauvais clustering, avec seulement 4 classes prédominantes et la classe labéllisée "1" qui prend beaucoup de volume sur l'espace comparée aux autres.
Par ailleurs, nous avons testé la méthode avec plusieurs matrices de similarités, et les meilleurs résultats sont obtenus avec la fonction corSimMat().

3.2 Réduction de la dimension avec t-SNE

Nous avons effectué une réduction de dimension en utilisant la méthode t-SNE (t-distributed Stochastic Neighbor Embedding) qui est une réduction de dimension non-linéaire contrairement à l'ACP.
Dans la figure 15, nous pouvons voir que la représentation en 2D de l'espace réduit donné par t-SNE est meilleur que celui obtenu avec l'ACP. De plus, les classes "s'assoir" et "s'allonger" sont très bien séparées, alors que les classes "se lever", "descendre", "marcher" et "monter" sont regroupées dans un cluster. On observe aussi que ces classes regroupées ont une forme plutôt spirale. On peut dire qu'on a ici deux clusters, l'un représentant les activitées statiques et l'autre les dynamiques.

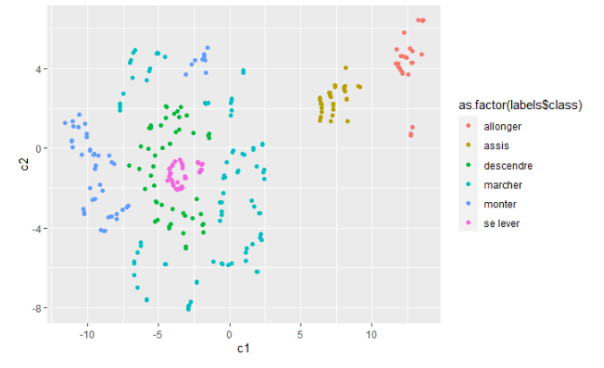

Figure 15: visualisation 2D de l'espace réduit t-SNE.

Après avoir réduit la dimension de données avec t-SNE, nous avons appliqué différents algorithmes de clustering comme nous le verrons dans ce qui suit.

3.2.1 K-means

Le k-means sur l'espace réduit t-SNE nous donne un résultat de NMI égal à 62% ce qui est meilleur que ce qu'a donné l'ACP.
La figure 16 ci dessous, montre le résultat de clustering K-means sur t-SNE ainsi que les différents clusters obtenus avec ce dernier.

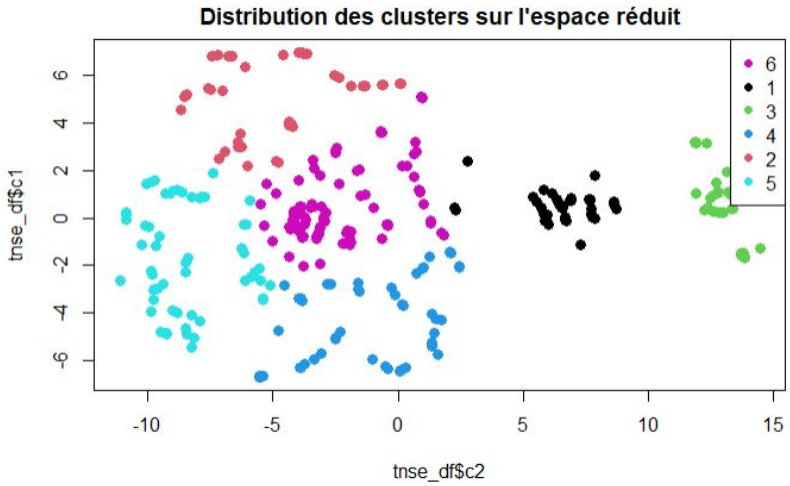

Figure 16: Visualisation des clusters obtenus avec K-means sur l'espace réduit avec t-SNE

On voit bien que sur le K-means a plus ou moins réussi à détecter la structure spirale des données de l'espace réduit avec t-SNE.

3.2.2 Clustering Spectral

Le clustering spectral sur l'espace réduit t-SNE a donné un taux de NMI atteignant les 61%, moins bon que le K-means mais meilleur que le résultat donné par le clustering spectral sur l'espace réduit avec l'ACP.
La figure 17, montre les clusters obtenus à l'issue de l'application du clustering spectral. On voit que le modèle a regroupé plusieures classes ensembles et a eu du mal à détecter la structure des données.

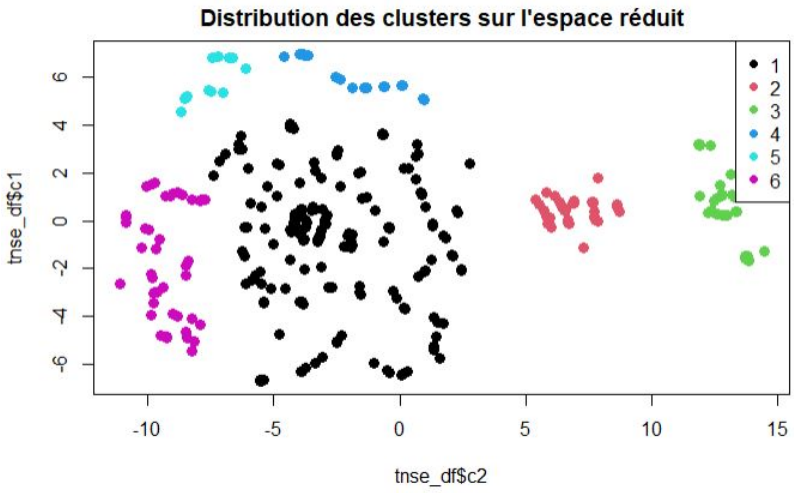

Figure 17: Visualisation des clusters obtenus avec le clustering spectral sur l'espace réduit avec t-SNE

3.2.3 Classification ascendante hiérarchique CAH

La classification ascendente hiérarchique a donné le meilleur résultat de clustering sur les données de l'espace réduit avec t-SNE avec un taux de NMI égal à 64
La figure 18, montre les clusters obtenus. La CAH a mieux détecté la structure des classes que le clustering spectral. En revanche les classes "se lever", "descendre" et "marcher" ont été mal séparées.

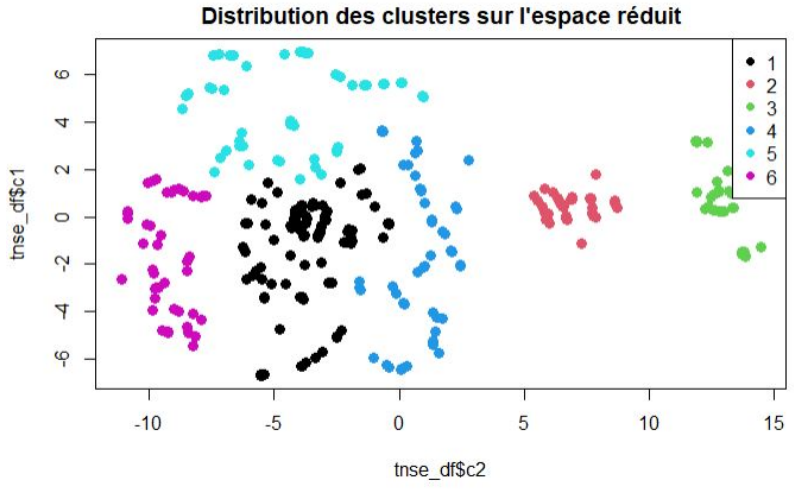

Figure 18: Visualisation des clusters obtenus avec le CAH sur l'espace réduit avec t-SNE

La figure 19 montre le dendrogramme obtenu à l'issu de la CAH sur les données réduites avec t-SNE.

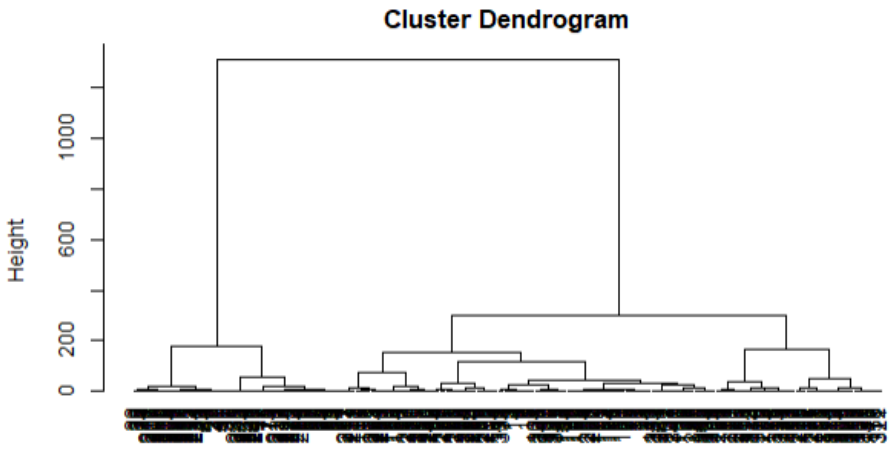

Figure 19: Dendrogramme de la CAH sur les données réduites avec t-SNE

3.2.4 DBSAN

La dernière méthode utilisé est DBSCAN. Ce modèle a donné un taux de NMI égal à 59%. Comparé aux méthodes vues précédemment DBSCAN donne le moins bon résultat de clustering.
La figure 20, montre les clusters obtenu à l'issue de la classification en utilisant DBSCAN.

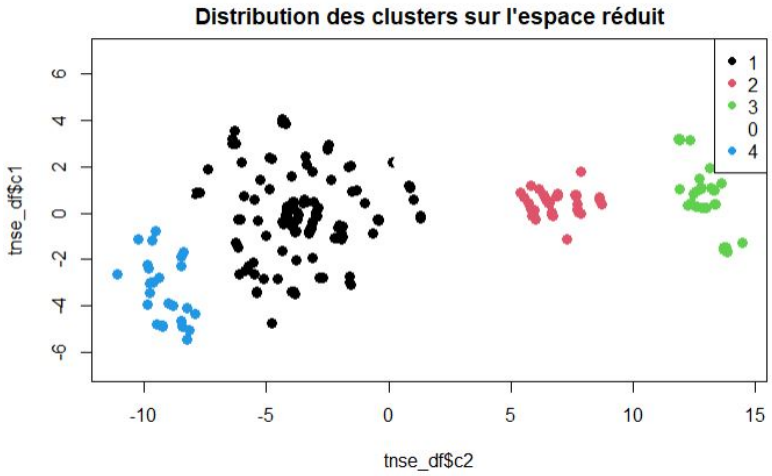

Figure 20: Résultats de classification avec DBSCAN sur les données réduites avec t-SNE

On voit bien que DBSCAN a eu du mal à bien séparer les classes ayant la structure spirale. On pourrait penser que cette mauvaise classificaton pourrait être dûe au fait que les observations ont une densité variée.

3.3 K-Medoids avec la distance DTW sur séries temporelles multivariées

Le package *dtwclust* comprend de nombreuses fonctionalités permettant de faire de l'analyse de séries temporelles. Il permet aussi de travailler avec des séries temporelles multivariées, ce qui est le cas ici. Nous utiliserons les 3 vitesses et les 3 accélérations estimées comme variables pour

l'algorithme.

La fonction utilisée est *tsclust()* qui fait du clustering de séries temporelles. Par défaut, cette fonction utilise comme distance une version spécifique du Dynamic Time Warping (DTW), moins précise mais beaucoup plus rapide à calculer que le DTW classique.

L'algorithme de clustering utilisé avec cette distance sera PAM (Partioning Around Medoids). En fixant k=6 et le nombre de répétitions à 10, nous obtenons les résultats suivants (figure 21) :

```
              [,1]        [,2]        [,3]        [,4]        [,5]        [,6]        [,7]        [,8]        [,9]
ARI     0.406064328 0.33881573 0.56509991 0.652210628 0.552623386 0.476493717 0.60265888 0.651212775 0.540830084
RI      0.824107544 0.77953058 0.85802335 0.887408172 0.857340374 0.831986807 0.86327064 0.890972997 0.865386217
J       0.345340691 0.31092831 0.48423601 0.564385151 0.471096838 0.408411051 0.52254086 0.560413728 0.452172734
FM      0.513434523 0.47978067 0.65785309 0.726628894 0.643784717 0.583406038 0.69808683 0.720519928 0.622753526
VI      0.656064936 0.64765232 0.55821369 0.392995714 0.626822970 0.657102480 0.36868709 0.448091030 0.435068935
Sil     0.062009533 0.25724321 0.28536473 0.139447743 0.194777623 0.064792880 0.40099572 0.256481199 0.058403239
SF      0.000000000 0.00000000 0.00000000 0.000000000 0.000000000 0.000000000 0.00000000 0.000000000 0.000000000
CH     56.500439386 64.01264240 51.57130961 74.728135485 51.262936547 55.073108453 77.63712832 74.442095050 81.580394907
DB     13.655196360 8.70832356 18.23146260 4.413889999 19.915433917 15.224534360 4.75171024 7.245111223 3.524116387
DBstar 21.977474873 14.43690598 24.45207608 18.840623680 30.178199404 22.787096807 9.37934716 20.284684226 14.903951159
D       0.007331329 0.01220494 0.01282977 0.008301799 0.008573959 0.007434796 0.01813448 0.008844746 0.007649587
COP     0.316184134 0.33232448 0.33043844 0.305793514 0.329198091 0.319787742 0.29843659 0.305562436 0.292565133
            [,10]
ARI     0.4780454
RI      0.8207926
J       0.4162778
FM      0.5975986
VI      0.3657928
Sil     0.4404163
SF      0.0000000
CH     84.4177077
DB      1.4771940
DBstar  1.8609710
D       0.2756110
COP     0.2868777
```

Figure 21: Scores pour les 10 répétitions de PAM

Nous choisissons la répétition n°8 qui a le meilleur ARI. La NMI associée est 0,676.

Si on affiche les moyennes des différentes variables pour chaque cluster prédit, nous obtenons le diagramme suivant (figure 23):

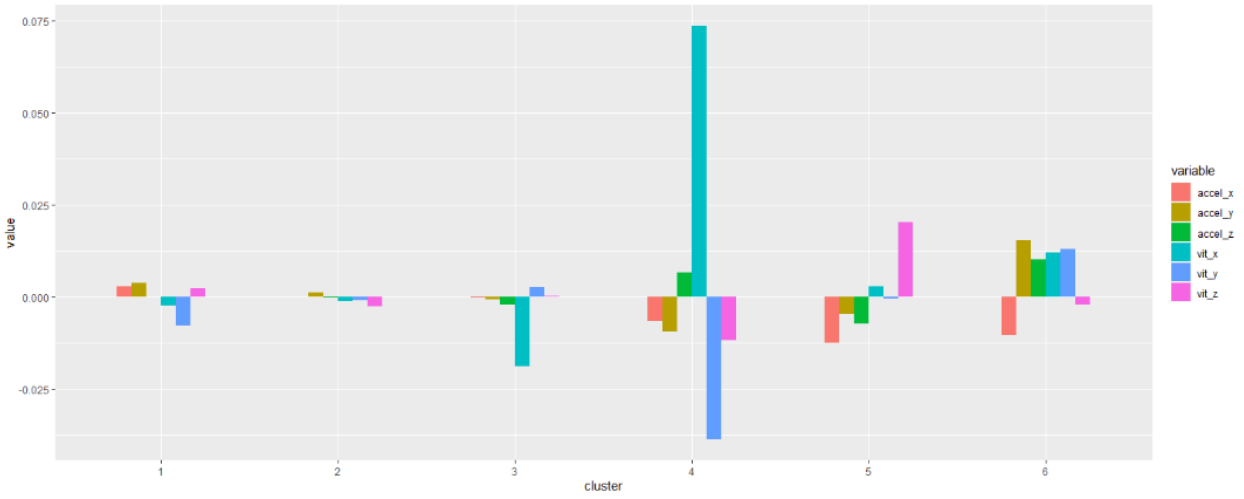

Figure 22: Moyenne par variable pour chaque cluster prédit

On ditingue bien des clusters correspondant à des phases actives, et d'autres à des phases plus passives où les moyennes des variables sont très basses.

Nous pouvons comparer ce diagramme avec celui pour les vrais clusters.

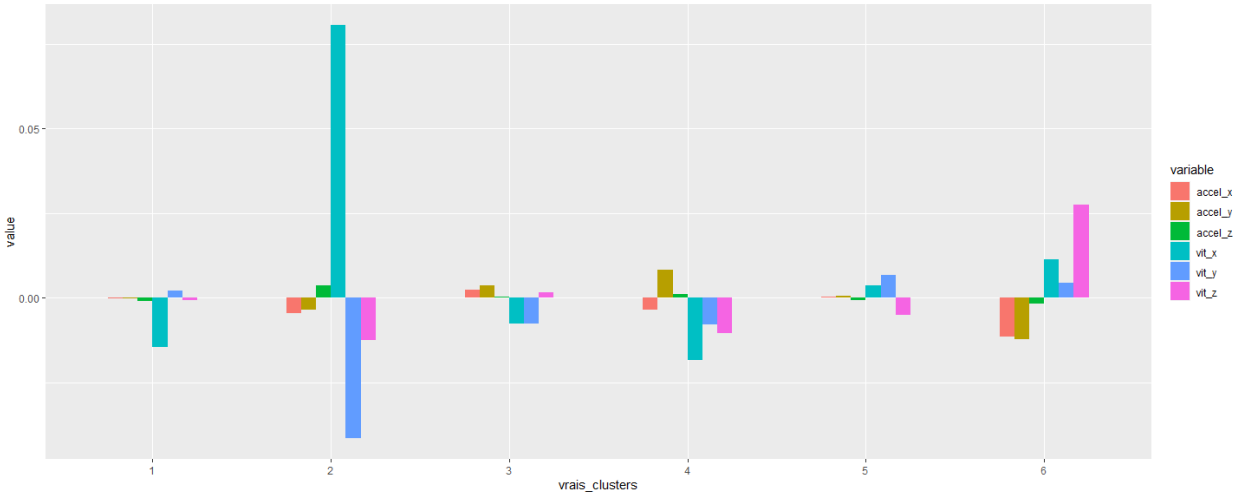

Figure 23: Moyenne par variable pour chaque véritable cluster

Certains clusters prédits ont des histogrammes quasiment identiques aux vrais clusters (notamment le cluster prédit n°4 qui a comme équivalent le vrai cluster n°2). Cela montre que notre clustering est plutôt de bonne qualité.

Distinguer chaque activité est plus compliqué, sachant qu'en faisant la moyenne, nous avons perdu la dimension temporelle des séries. Nous pouvons cependent penser que les clusters avec des vitesses en z positives et non négligeables correspondent à des phases ascendantes (monter un escalier, se lever), et celles avec des vitesses z négatives correspondent à des phases descendantes.

4 Conclusion

Dans ce projet, nous avons pu expérimenter différentes méthodes de classification sur des données temporelles.
Nous avons testé ces méthodes sur les données d'origine, mais aussi sur des espaces réduits avec l'ACP et t-SNE.
Nous avons réussi dans plusieurs cas à obtenir d'assez bons résultats qui se rapprochent de la réalité.
Nous avons également remarqué que certains modèles distinguent beaucoup plus facilement les deux phases d'activité (dynamique et statique), mais trouvent plus de difficulté à séparer les 6 activités.
Nous avons obtenu le meilleur score grâce à la classification sur les séries temporelles multivariées avec la librairie tsclust qui nous donne un score NMI de 68% environ.
Par contre, sur les espaces réduits, on a remarqué que t-SNE donne une meilleures représentations de données, ce qui permet d'avoir de meilleurs résultats qu'avec l'ACP par exemple ou l'espace d'origine.