olist



# Segmentation des clients olist

Marc Sellam

Projet 5 Data Scientist 09/2022

# Sommaire

# Introduction et contexte

Olist opère dans le segment du e-commerce, et concentre les produits de tous les vendeurs dans un seul magasin visible par le consommateur final.

Olist souhaite une segmentation clients, et un conseil de fréquence à laquelle la segmentation doit être mise à jour.

# Etude exploratoire

Olist fournit une base de données débutant en janvier 2017.

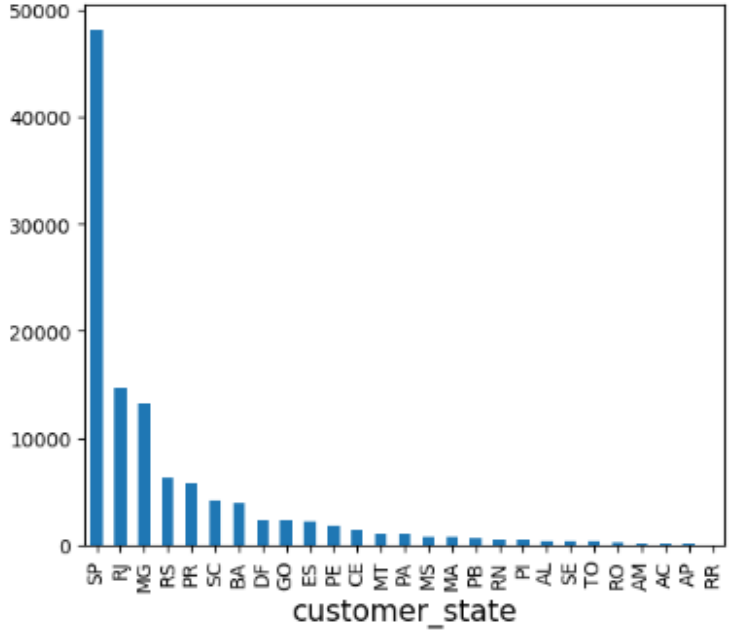
Les données fournies se composent de 9 fichiers au formats csv.

Ces fichiers vont être fusionné afin d'en synthétiser un jeu de données unique de 95568 commandes contenant:

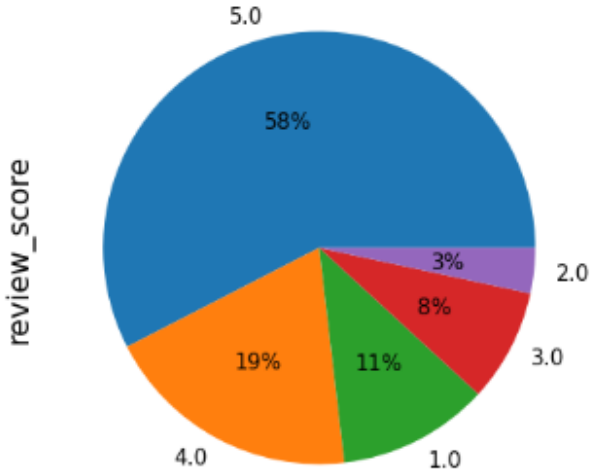
- Le numéro de commande.
- L'identification client attribuée lors de la commande.
- L'identification client unique.
- Les quantités achetées.
- Le prix total de la commande.
- La moyenne du score de satisfaction de la commande.
- La date de la commande.

# Les variables catégorielles

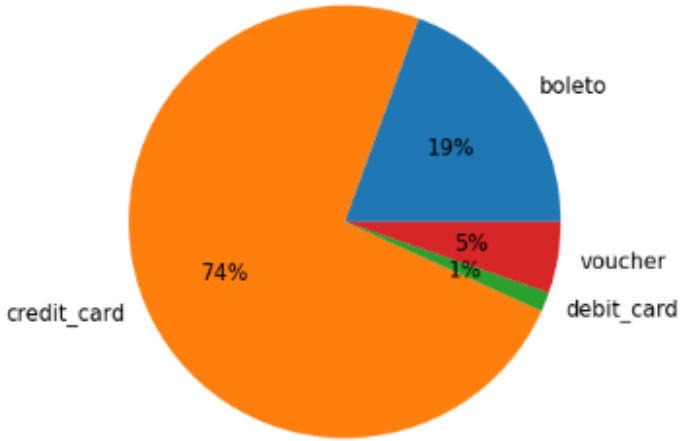
Fréquences régions d'achat




Proportions des review score

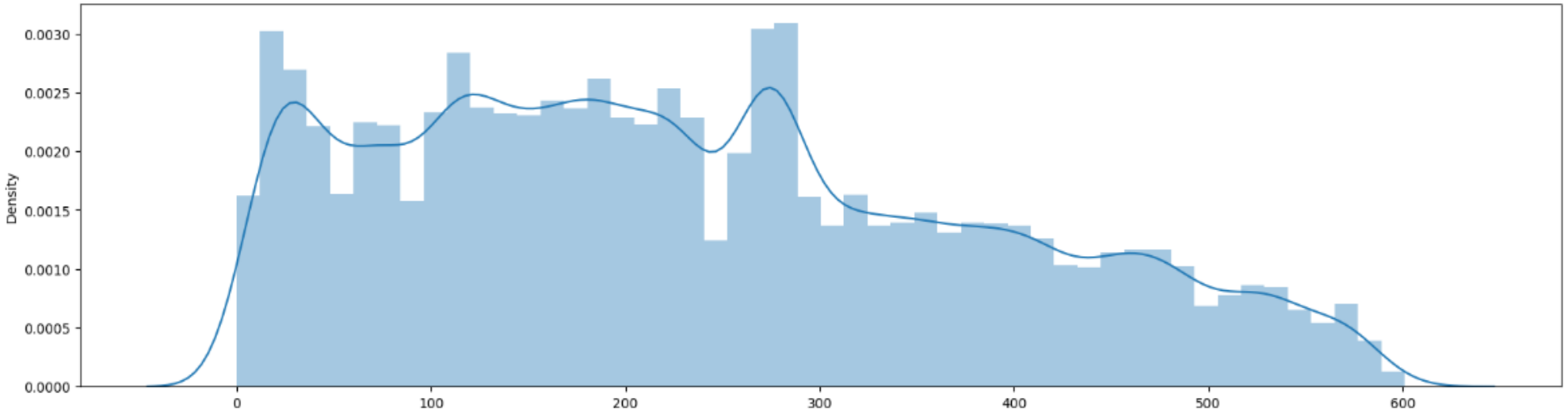



Proportions du mode de règlement



# Les variables numériques

| | note_moy | Monetary | qtés | Recency | Frequency |
|---|---|---|---|---|---|
| **count** | 92504.000000 | 92504.000000 | 92504.000000 | 92504.000000 | 92504.000000 |
| **mean** | 4.154123 | 147.600478 | 1.230725 | 236.099563 | 1.033123 |
| **std** | 1.279691 | 242.754956 | 0.816245 | 150.962701 | 0.208324 |
| **min** | 1.000000 | 0.850000 | 1.000000 | 0.000000 | 1.000000 |
| **25%** | 4.000000 | 48.900000 | 1.000000 | 114.000000 | 1.000000 |
| **50%** | 5.000000 | 89.900000 | 1.000000 | 218.000000 | 1.000000 |
| **75%** | 5.000000 | 159.770000 | 1.000000 | 344.000000 | 1.000000 |
| **max** | 5.000000 | 13440.000000 | 75.000000 | 601.000000 | 15.000000 |



Fréquence du nombre de clients par recence en jours(derniere commande).

# Les variables numériques

| Frequency | note_moy | Moyenne_depensé | qtés_moyenne | Recence_moyenne | nb_clients |
|---|---|---|---|---|---|
| 1 | 4.152487 | 143.772819 | 1.184854 | 236.616220 | 89752 |
| 2 | 4.188364 | 257.436997 | 2.564204 | 220.843935 | 2531 |
| 3 | 4.418902 | 378.700747 | 4.017241 | 211.551724 | 174 |
| 4 | 4.446429 | 714.675357 | 5.821429 | 173.750000 | 28 |
| 5 | 4.300000 | 624.060000 | 5.666667 | 126.888889 | 9 |
| 6 | 4.666667 | 520.132000 | 8.200000 | 212.600000 | 5 |
| 7 | 5.000000 | 775.340000 | 10.333333 | 115.666667 | 3 |
| 9 | 2.777778 | 1000.850000 | 14.000000 | 183.000000 | 1 |
| 15 | 5.000000 | 714.630000 | 15.000000 | 9.000000 | 1 |

89752 clients : 1 commande

2752 clients : plus d'une commande

92504 clients au total

Du 5 janvier 2017 au 29 aout 2018.

**dates premiere et derniere commande du fichier**

```
order_items['order_purchase_timestamp'] = pd.to_datetime(
    order_items['order_purchase_timestamp'])

order_items.order_purchase_timestamp.min(
), order_items.order_purchase_timestamp.max()
```

```
(Timestamp('2017-01-05 11:56:06'), Timestamp('2018-08-29 15:00:37'))
```

# Essais RFM:

Segmentation RFM :

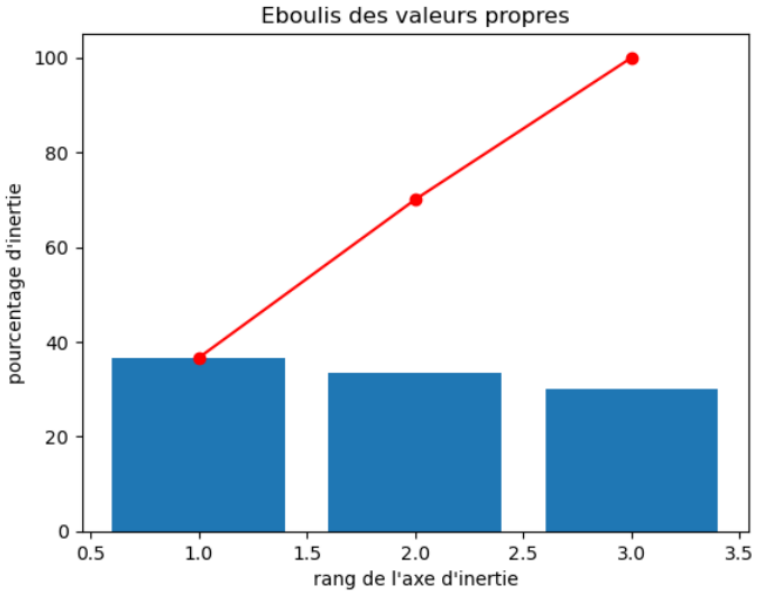
- Récence
- Fréquence
- Montant

| customer_unique_id | Recency | Frequency | Monetary |
|---|---|---|---|
| 0000366f3b9a7992bf8c76cfdf3221e2 | 111 | 1 | 129.90 |
| 0000b849f77a49e4a4ce2b2a4ca5be3f | 114 | 1 | 18.90 |
| 0000f46a3911fa3c0805444483337064 | 537 | 1 | 69.00 |
| 0000f6ccb0745a6a4b88665a16c9f078 | 321 | 1 | 25.99 |
| 0004aac84e0df4da2b147fca70cf8255 | 288 | 1 | 180.00 |

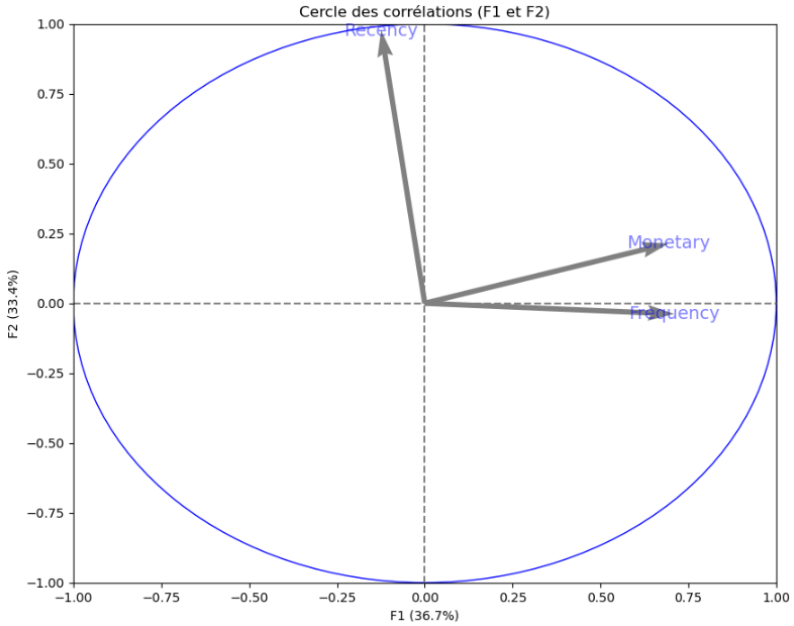
```
rfm.describe()
```

| | Recency | Frequency | Monetary |
|---|---|---|---|
| count | 92504.000000 | 92504.000000 | 92504.000000 |
| mean | 236.099563 | 1.033123 | 147.600478 |
| std | 150.962701 | 0.208324 | 242.754956 |
| min | 0.000000 | 1.000000 | 0.850000 |
| 25% | 114.000000 | 1.000000 | 48.900000 |
| 50% | 218.000000 | 1.000000 | 89.900000 |
| 75% | 344.000000 | 1.000000 | 159.770000 |
| max | 601.000000 | 15.000000 | 13440.000000 |

# ACP

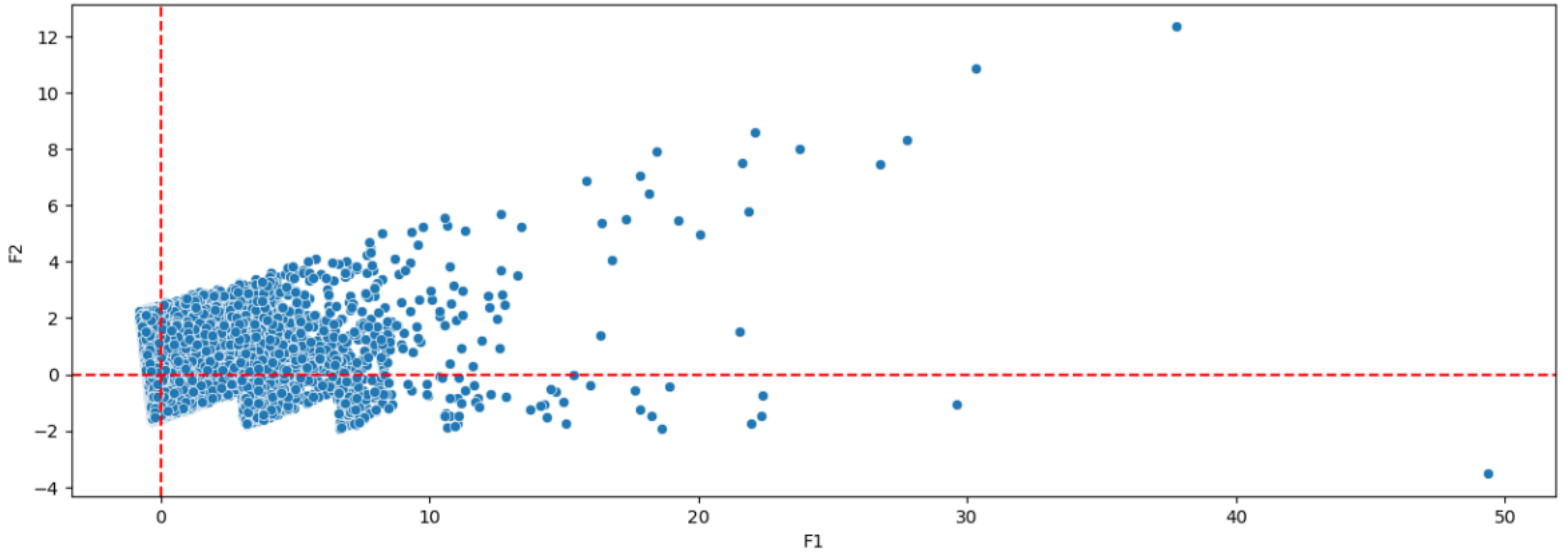



Le cumul de la variance expliqué est de de 70% sur les deux premiers facteurs.



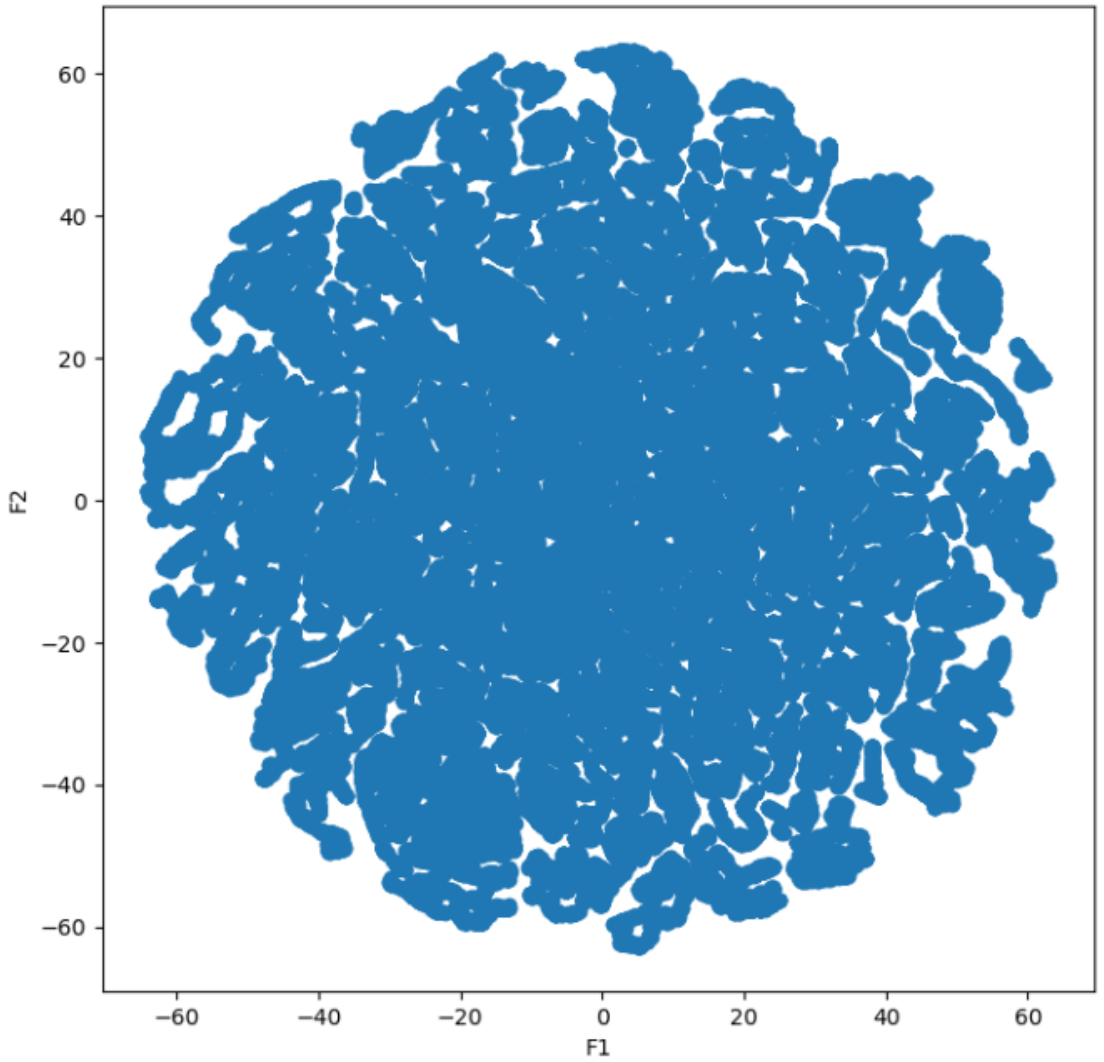

La variable Frequency est corrélée à F1, Monetary également.

La variable Recency est très bien représentée et corrélée à F2



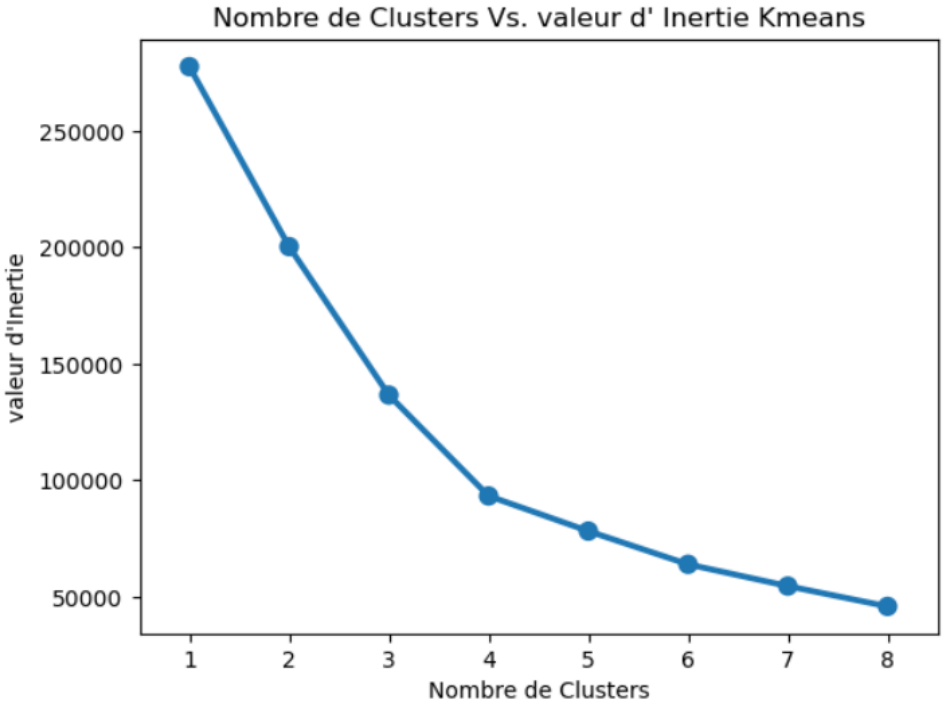
# t-SNE

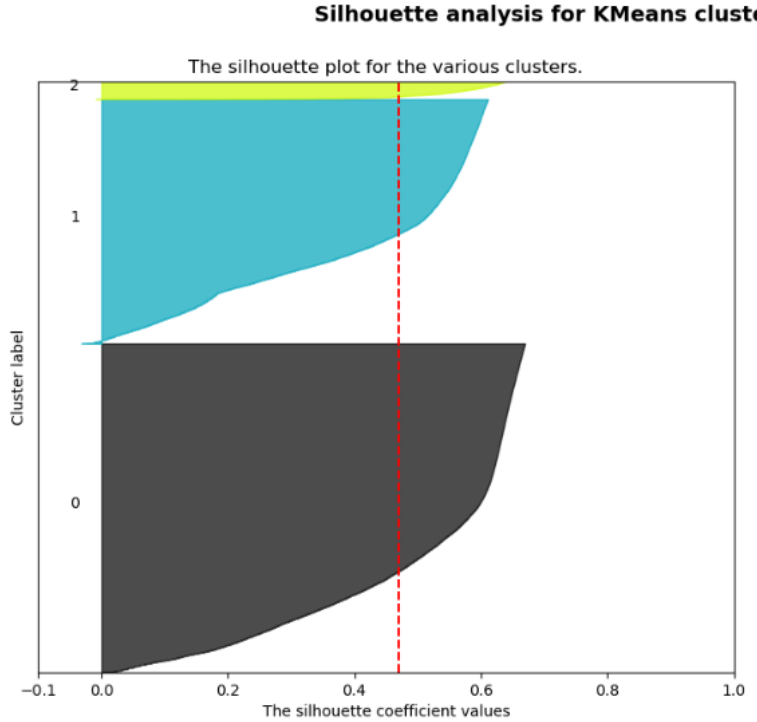
Résultat obtenu par réduction de dimension avec la **méthode t-SNE**

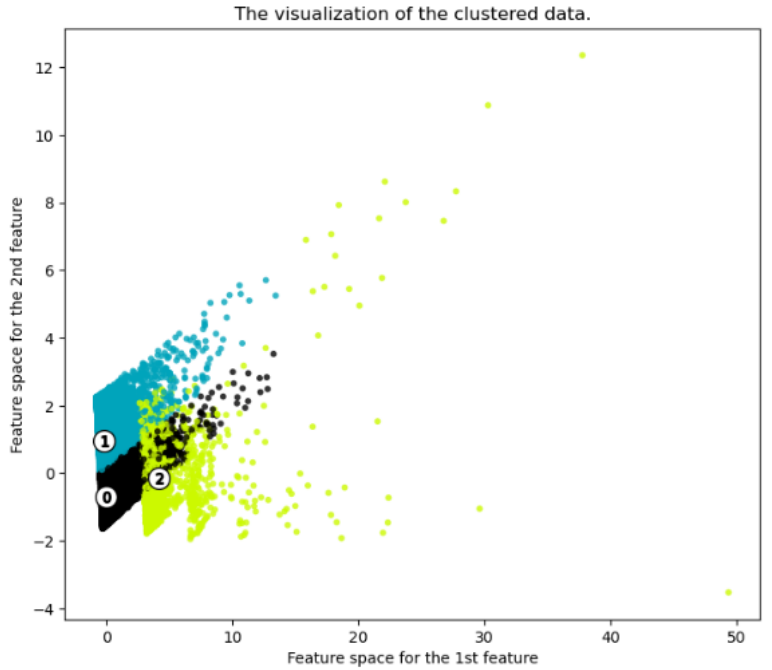



temps:_____ 0:04:48.066836

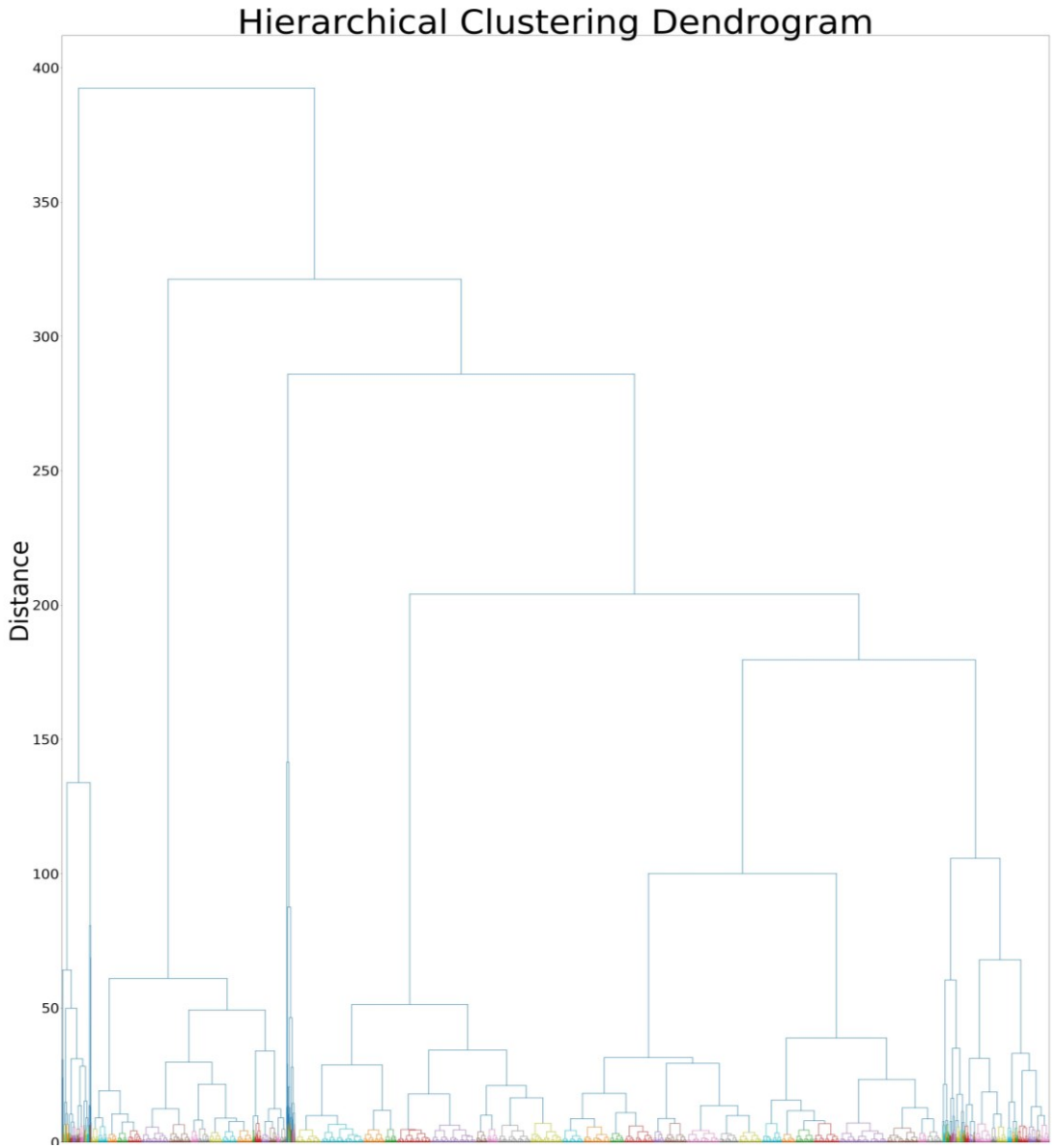
# KMeans


Nombre de Clusters Vs. valeur d' Inertie Kmeans
valeur d'Inertie
250000
200000
150000
100000
50000
1
2
3
4
5
6
7
8
Nombre de Clusters


Silhouette analysis for KMeans clustering on sample data with n_clusters = 3
The silhouette plot for the various clusters.
Cluster label
2
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
1.0
The silhouette coefficient values


The visualization of the clustered data.
Feature space for the 2nd feature
12
10
8
6
4
2
0
−2
−4
0
10
20
30
40
50
Feature space for the 1st feature
1
2
0

# Dendrogramme




4 groupes

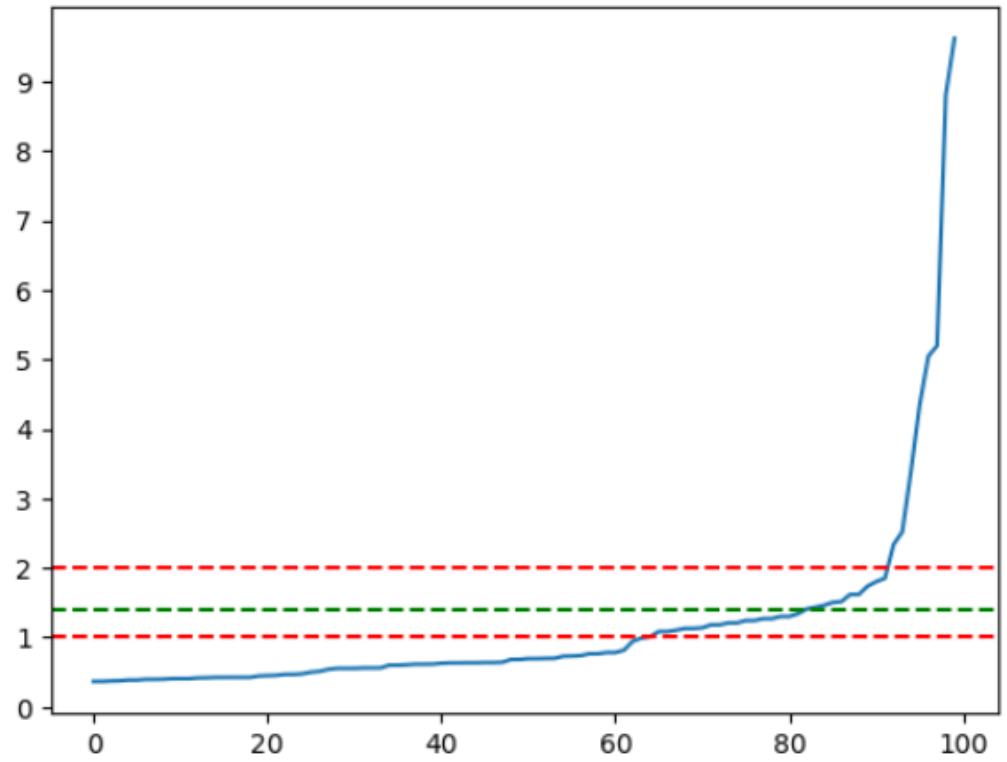
temps:_____ 2:18:58.300968

# DBscan

```
len(distances)
```

```
92504
```

```
distances = np.sort(distances, axis=0)
distances = distances[92402:92502,1]
plt.plot(distances)
plt.yticks(np.arange(0, 10, 1))
plt.axhline(y=1, color='r', linestyle='--')
plt.axhline(y=1.4, color='g', linestyle='--')
plt.axhline(y=2, color='r', linestyle='--');
#La valeur optimale pour epsilon sera trouvée au point de courbure maximale.
```

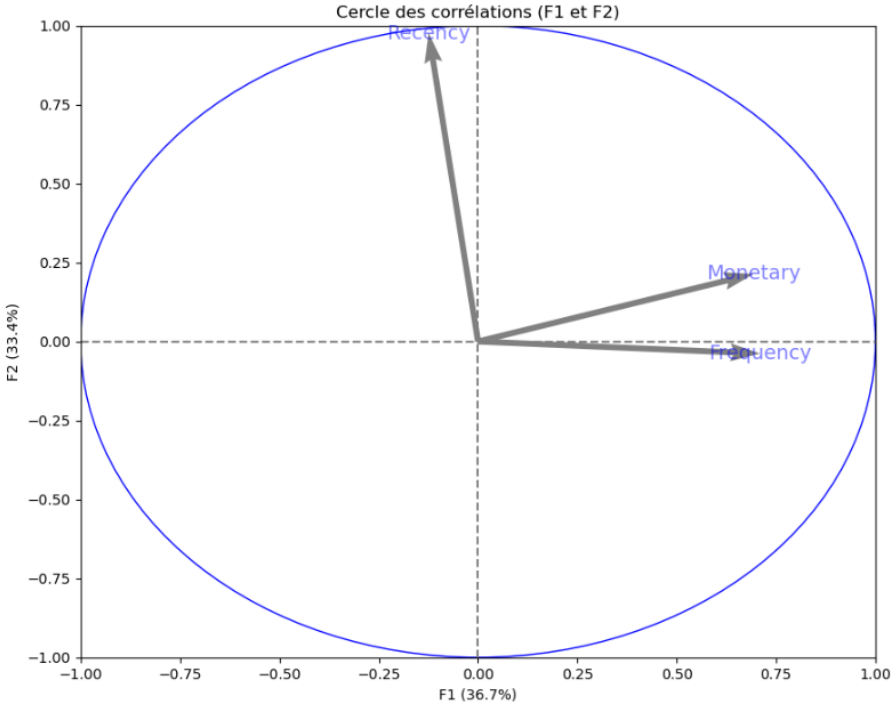



Choisir un **ε** de tel sorte que le maximum des observations aient une distance au proche voisin inférieure à **ε**

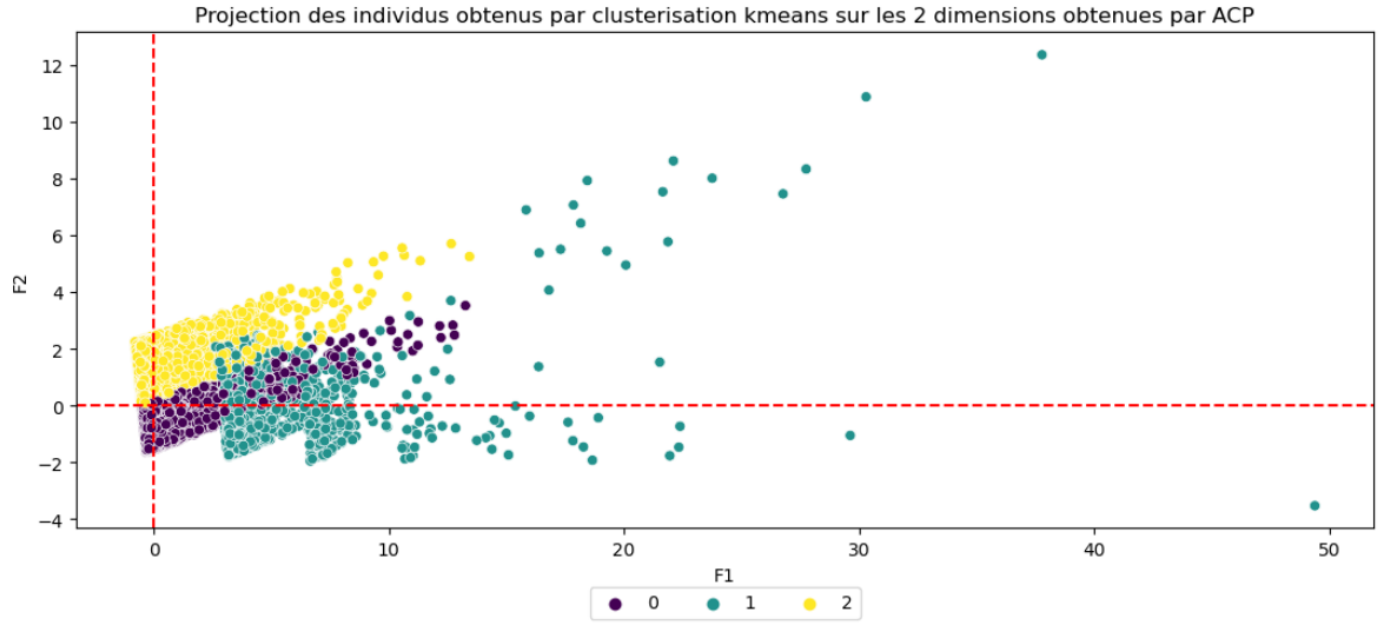
```
debut = datetime.datetime.now()

X = rfm_normalized.values

epsilon = 1.4
min_samples = 10

# Compute DBSCAN
db = DBSCAN(eps=epsilon, min_samples=min_samples,
            metric='euclidean', n_jobs=-1).fit(X)
labels = db.labels_

no_clusters = len(np.unique(labels))
no_noise = np.sum(np.array(labels) == -1, axis=0)

print('Estimated no. of clusters: %d' % no_clusters)
print('Estimated no. of noise points: %d' % no_noise)

fin = datetime.datetime.now()
print("temps:_________________", fin-debut)
```

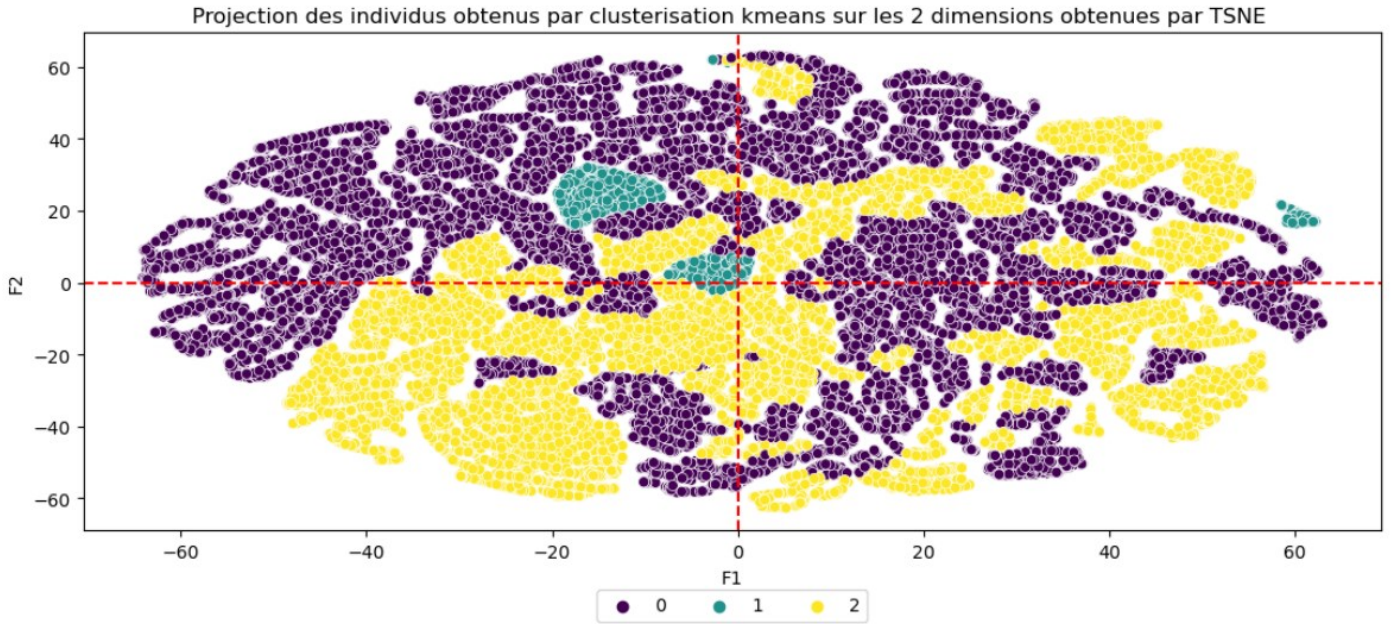
```
Estimated no. of clusters: 5
Estimated no. of noise points: 70
temps:_________________ 0:05:47.208724
```

# Résultats KMeans

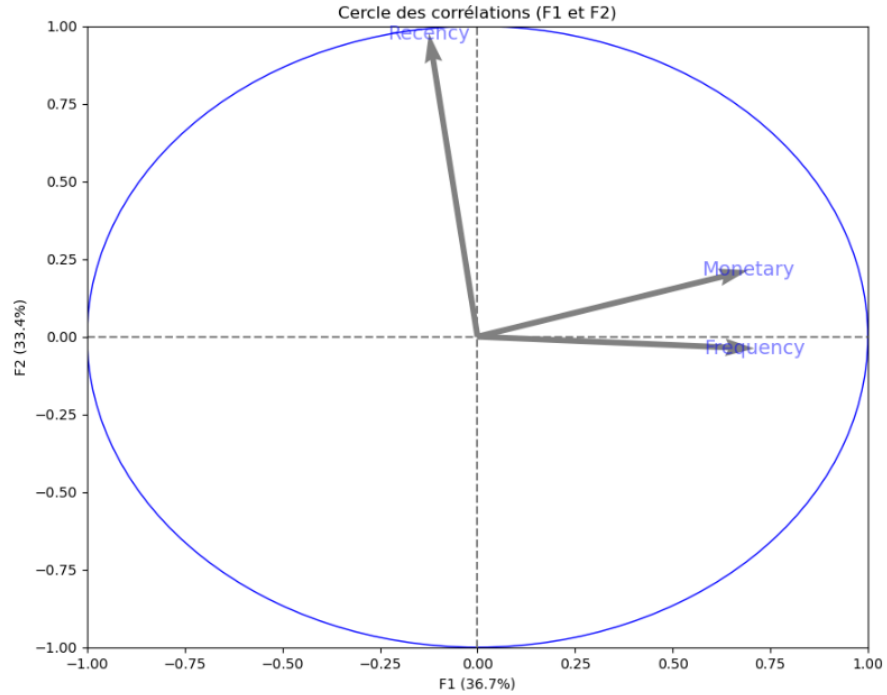



| km_cluster | Recency_m | Frequency_m | Monetary_m | nb_clients |
|---|---|---|---|---|
| 0 | 126.939824 | 1.000000 | 140.632719 | 51466 |
| 1 | 219.647633 | 2.107336 | 313.825688 | 2767 |
| 2 | 384.084659 | 1.000000 | 144.952456 | 38271 |

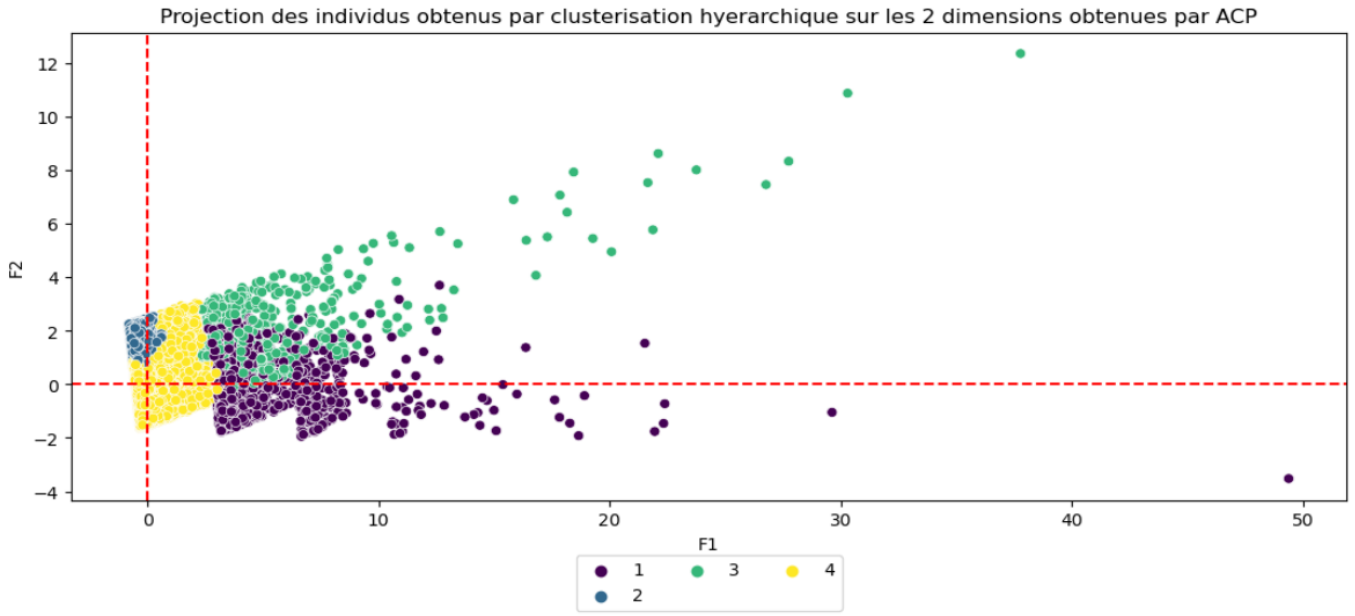


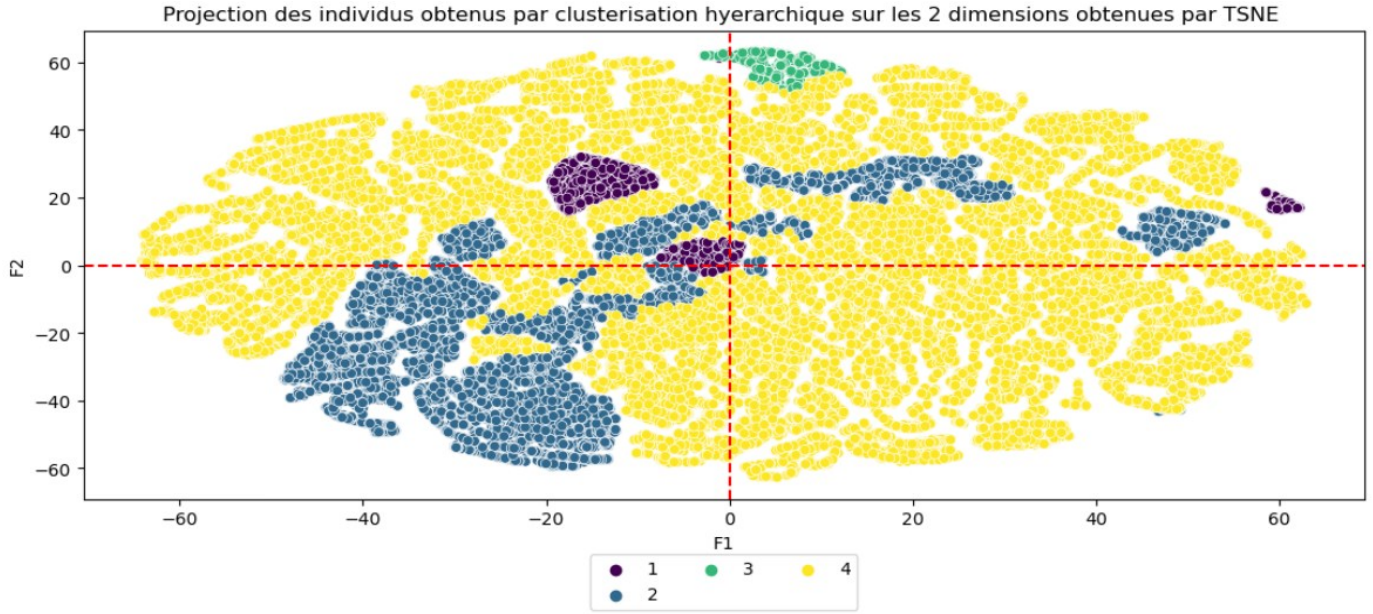



# Résultats Clusterisation Hiérarchique

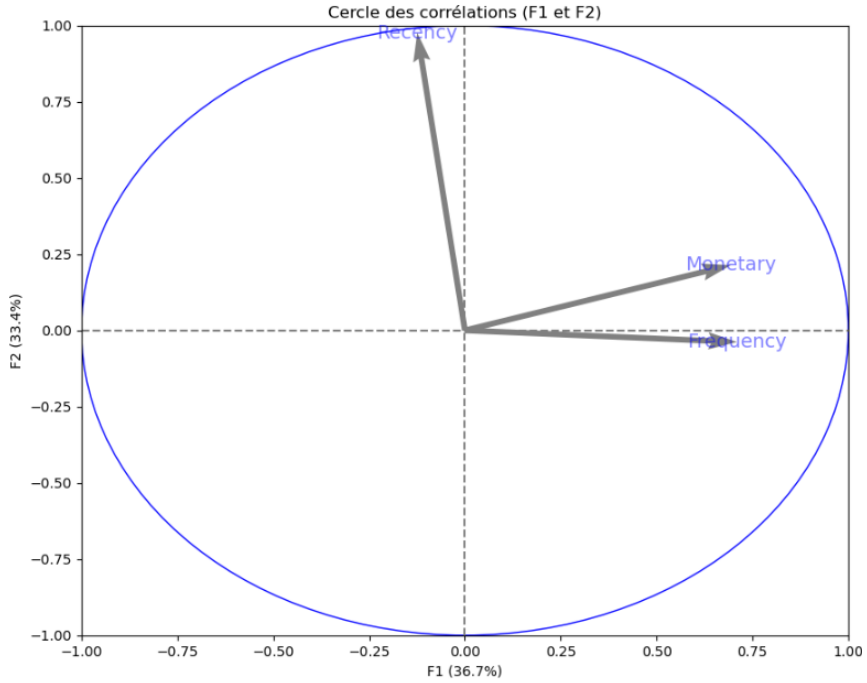



| h_cluster | Recency_m | Frequency_m | Monetary_m | Nb_clients |
|---|---|---|---|---|
| 1 | 219.026909 | 2.113455 | 267.968029 | 2750 |
| 2 | 461.758122 | 1.000000 | 111.420522 | 18315 |
| 3 | 266.244114 | 1.002478 | 1864.658302 | 807 |
| 4 | 177.906204 | 1.000000 | 132.677490 | 70632 |

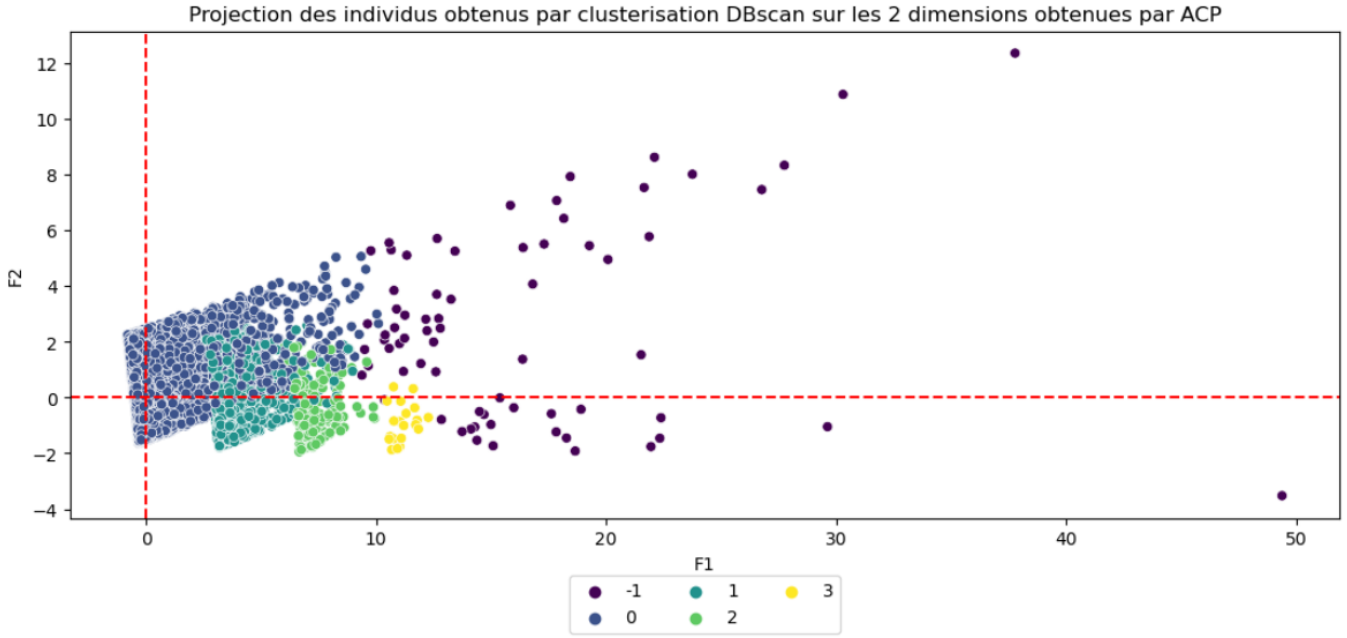


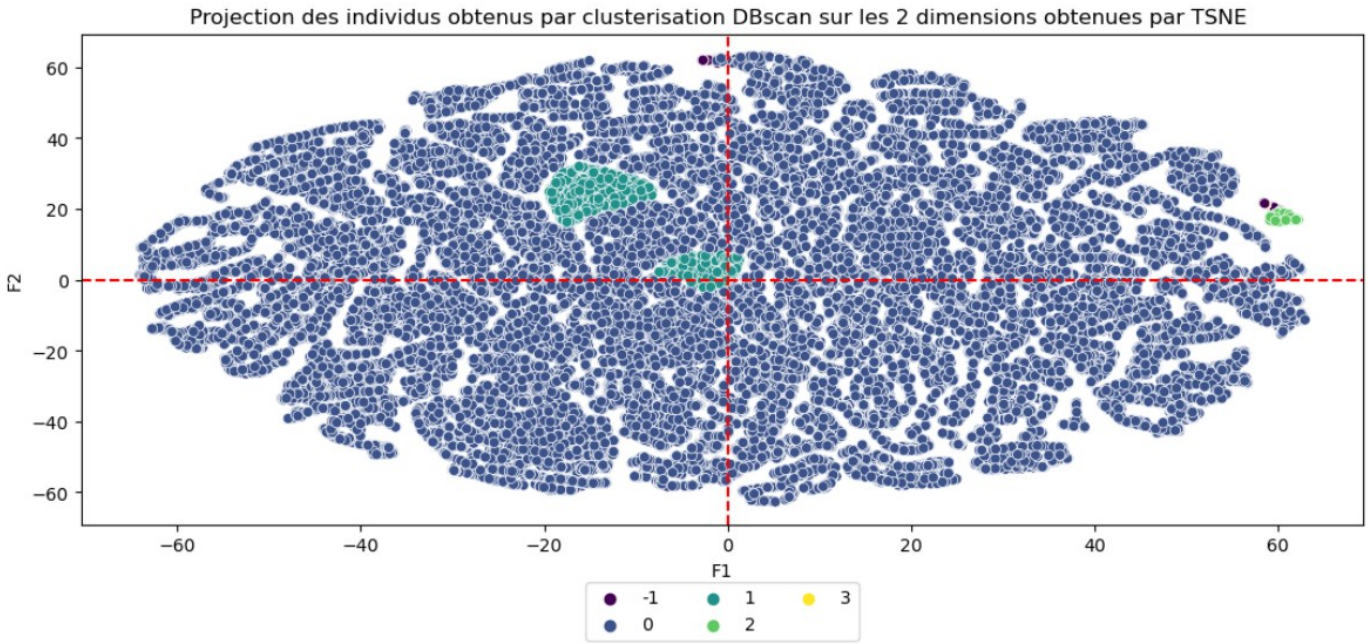



# Résultats DBscan




| db_cluster | Recency_m | Frequency_m | Monetary_m | Nb_clients |
|---|---|---|---|---|
| -1 | 237.471429 | 2.985714 | 3544.941429 | 70 |
| 0 | 236.608299 | 1.000000 | 141.657692 | 89719 |
| 1 | 220.462574 | 2.000000 | 248.634986 | 2525 |
| 2 | 211.341176 | 3.000000 | 344.755824 | 170 |
| 3 | 133.750000 | 4.000000 | 484.520500 | 20 |



# Etude finale :

## Sélection des variables

```
num_data = data_sli('2018-8-30')
print(num_data.shape)
num_data.head()
```

```
(92504, 4)
```

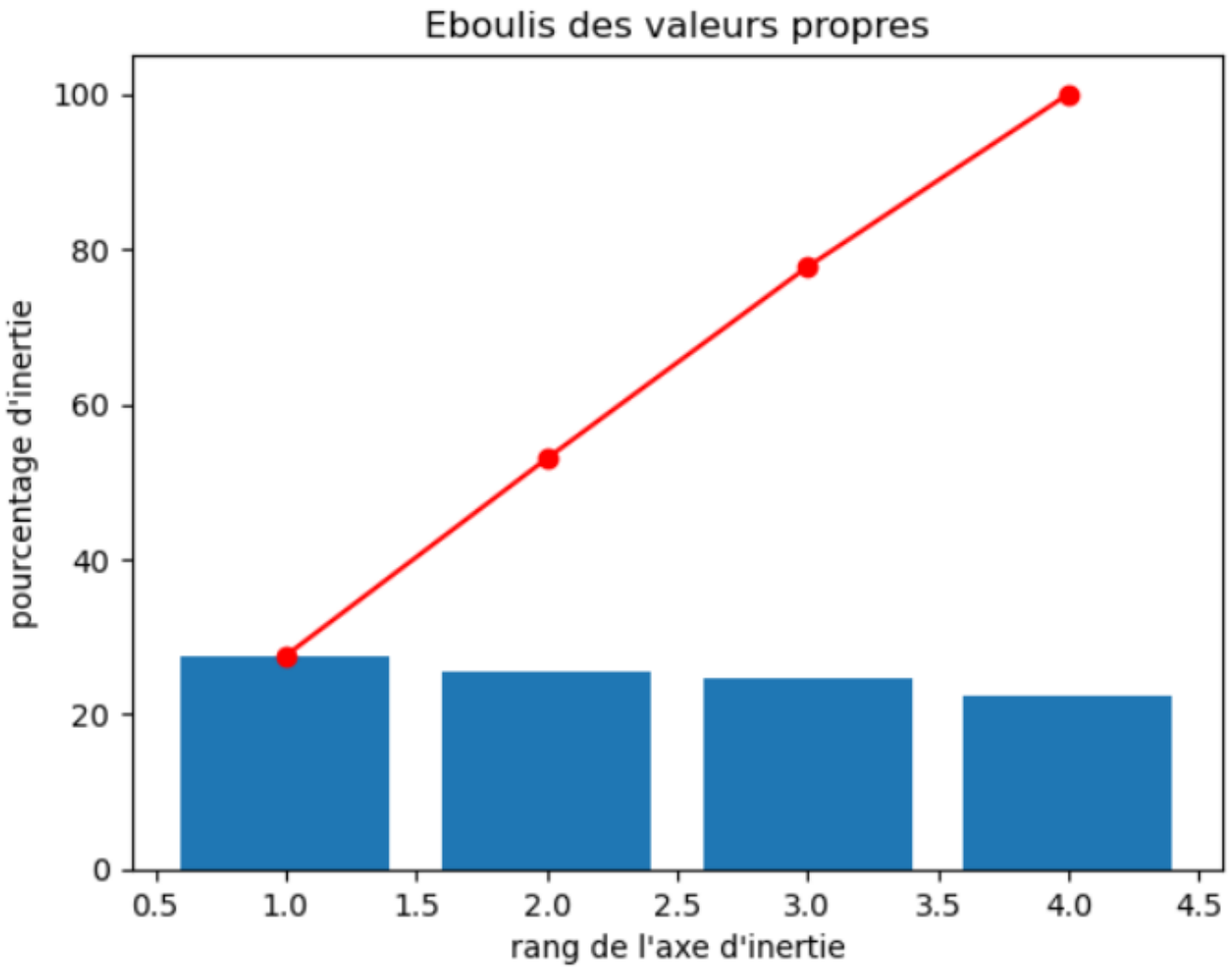
| customer_unique_id | note_moy | Monetary | Recency | Frequency |
|---|---|---|---|---|
| 0000366f3b9a7992bf8c76cfdf3221e2 | 5.0 | 129.90 | 111 | 1 |
| 0000b849f77a49e4a4ce2b2a4ca5be3f | 4.0 | 18.90 | 114 | 1 |
| 0000f46a3911fa3c0805444483337064 | 3.0 | 69.00 | 537 | 1 |
| 0000f6ccb0745a6a4b88665a16c9f078 | 4.0 | 25.99 | 321 | 1 |
| 0004aac84e0df4da2b147fca70cf8255 | 5.0 | 180.00 | 288 | 1 |

```
num_data.describe()
```

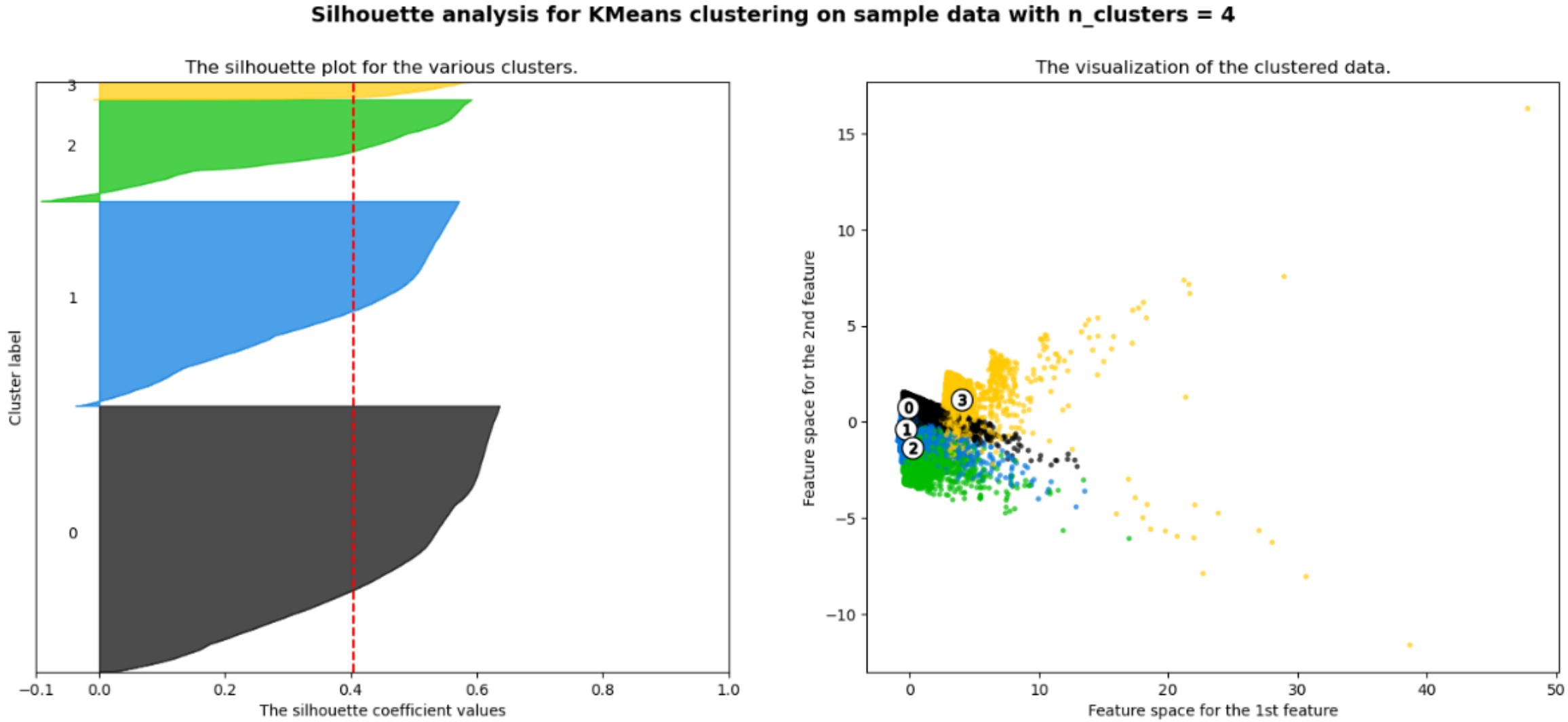
| | note_moy | Monetary | Recency | Frequency |
|---|---|---|---|---|
| count | 92504.000000 | 92504.000000 | 92504.000000 | 92504.000000 |
| mean | 4.154123 | 147.600478 | 236.099563 | 1.033123 |
| std | 1.279691 | 242.754956 | 150.962701 | 0.208324 |
| min | 1.000000 | 0.850000 | 0.000000 | 1.000000 |
| 25% | 4.000000 | 48.900000 | 114.000000 | 1.000000 |
| 50% | 5.000000 | 89.900000 | 218.000000 | 1.000000 |
| 75% | 5.000000 | 159.770000 | 344.000000 | 1.000000 |
| max | 5.000000 | 13440.000000 | 601.000000 | 15.000000 |

# ACP

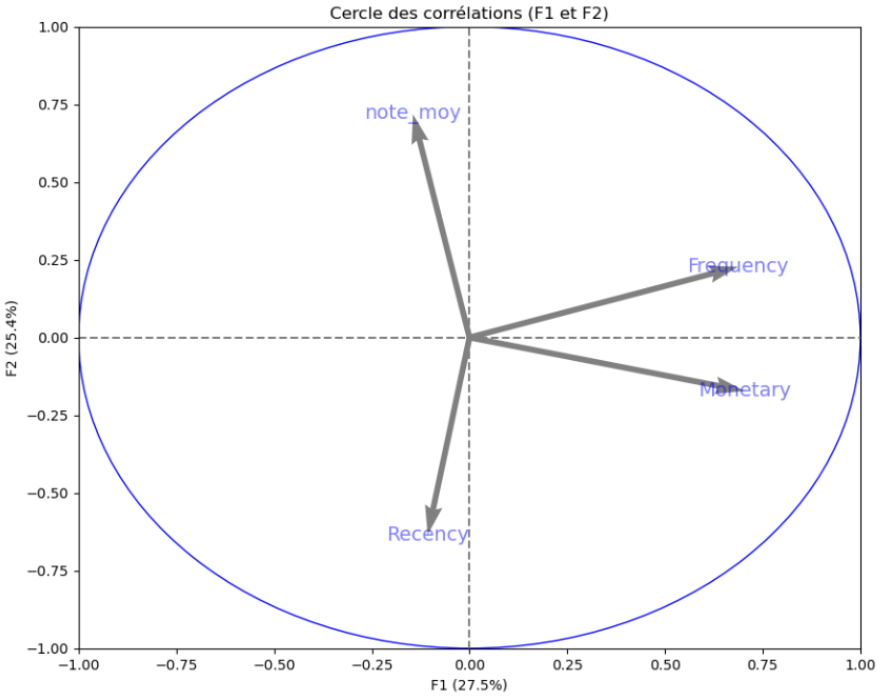
Le cumul de la variance expliqué est de de 53% sur les deux premiers facteurs.

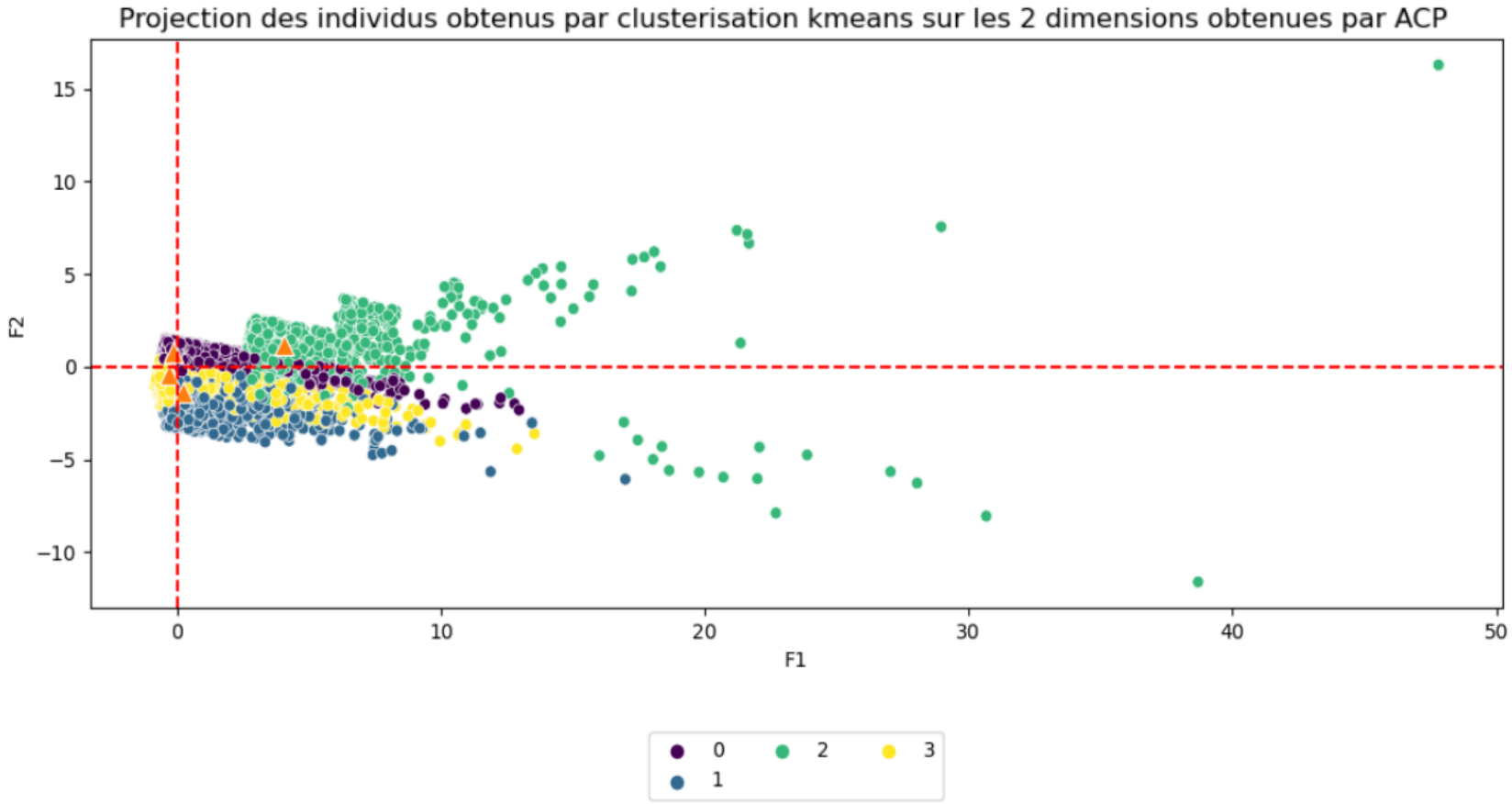


# KMeans


Silhouette analysis for KMeans clustering on sample data with n_clusters = 4
The silhouette plot for the various clusters.
Cluster label
3
2
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
1.0
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
15
10
5
0
−5
−10
0
10
20
30
40
50
Feature space for the 1st feature
0
1
2
3

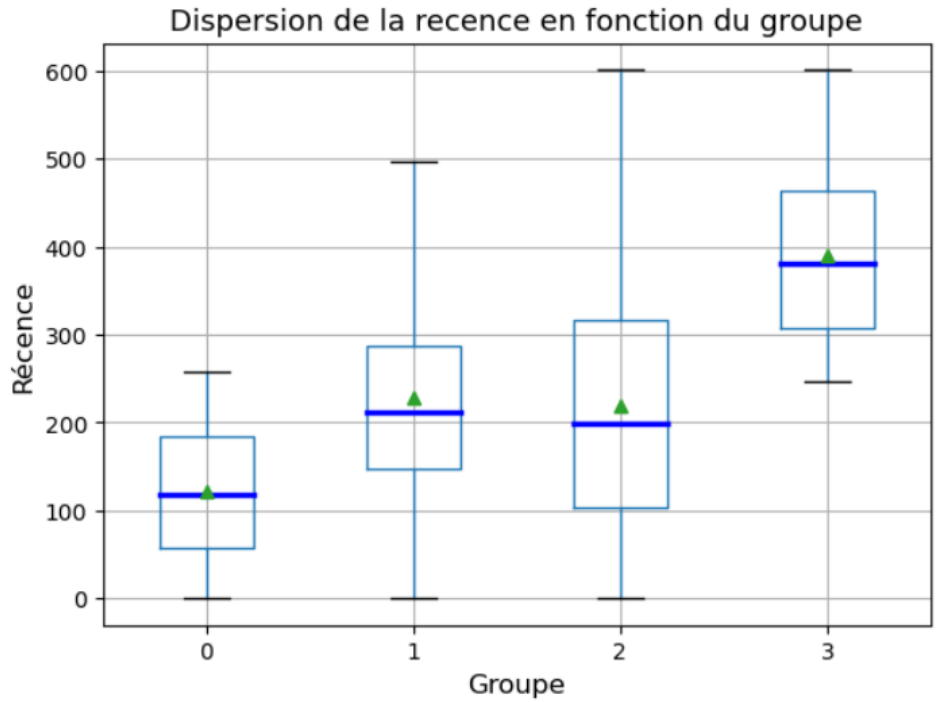
# Projection KMeans

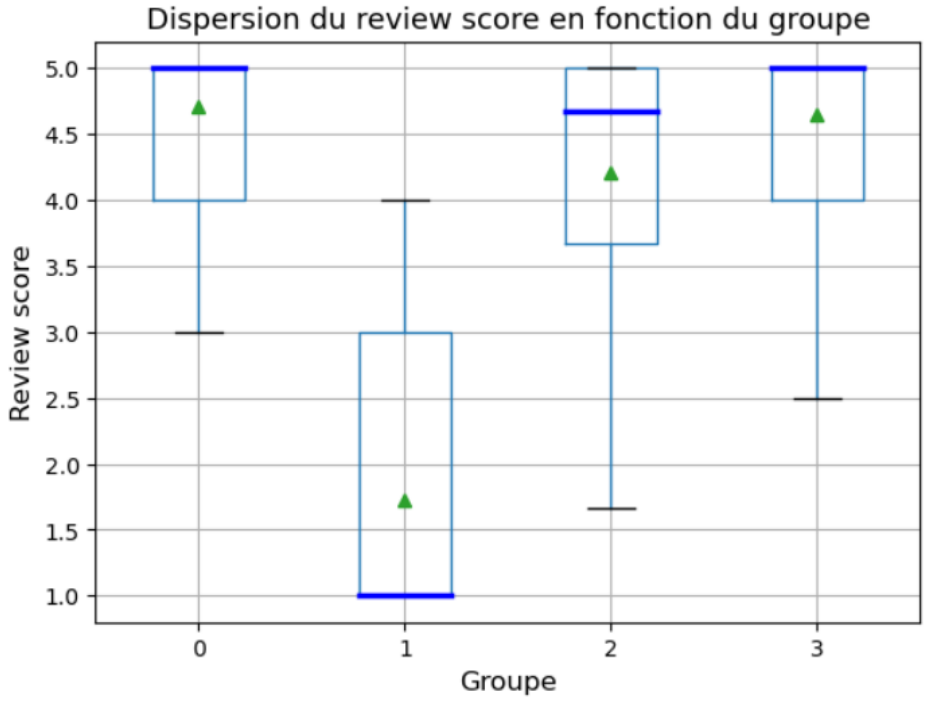


# Résultat

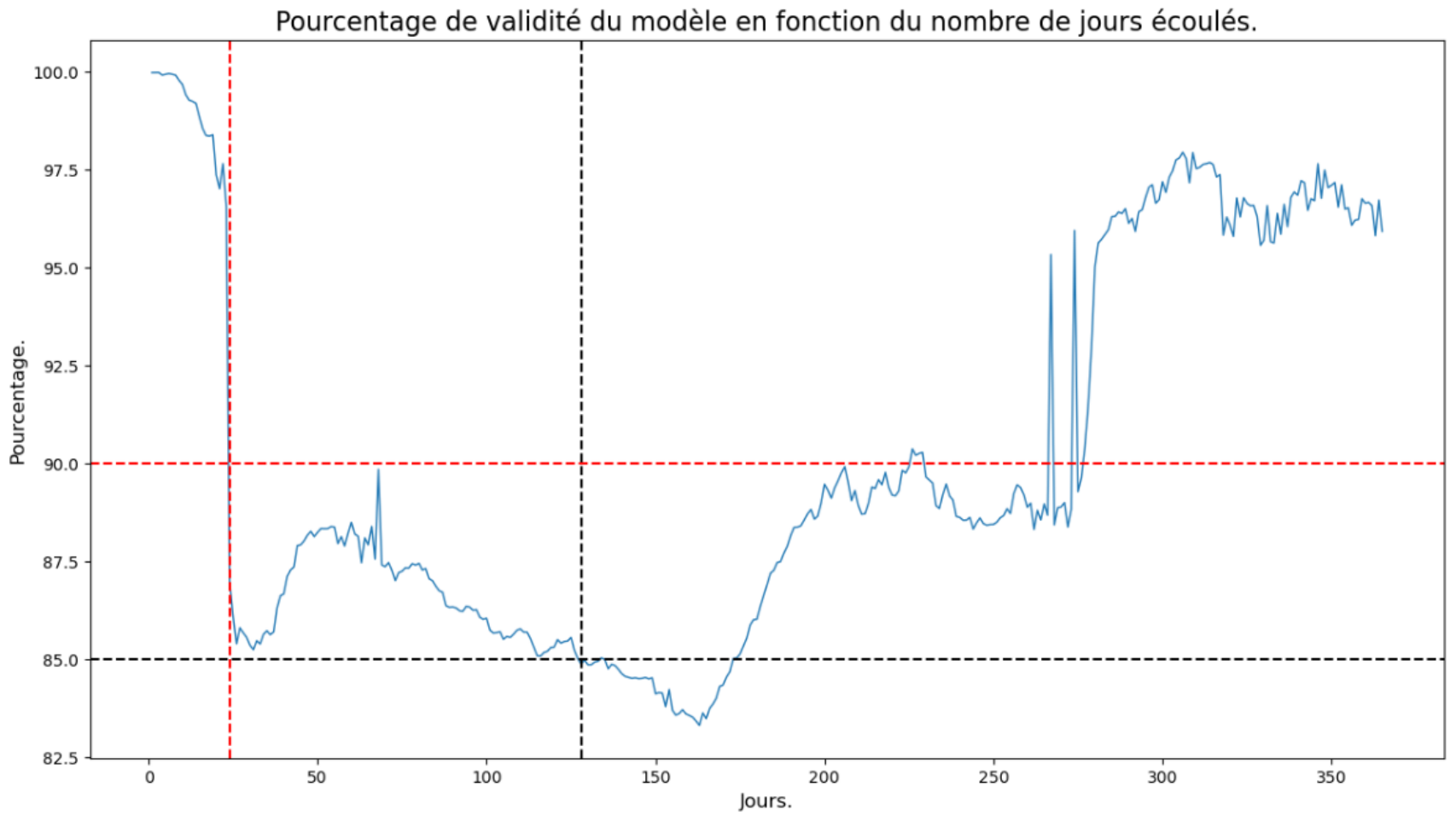
| km_cluster | Recency_m | Frequency_m | Monetary_m | Note_moy | Nb_clients |
|---|---|---|---|---|---|
| 0 | 121.509812 | 1.000000 | 136.252040 | 4.703474 | 41735 |
| 1 | 229.112778 | 1.000000 | 159.875650 | 1.725430 | 15934 |
| 2 | 219.626537 | 2.107737 | 311.793594 | 4.206056 | 2766 |
| 3 | 390.120428 | 1.000000 | 142.108456 | 4.641445 | 32069 |



# Maintenance

```
Nb de jours pour une maintenance a 90% de validité du modèle : 24 .
Nb de jours pour une maintenance a 85% de validité du modèle : 128 .
```



# Conclusion

L’analyse en composante principale et Kmeans se sont révélés être les algorithmes les plus adaptés à la résolution de notre problématique et à la création de notre modèle.