Parcours Data Scientist : Projet 5

Segmentez des clients d'un site e-commerce

Problématique

- Proposer une segmentation de la base de donnée client
- Evaluer fréquence de mise à jour de la segmentation

Interprétation/Piste de recherche

- Définir des indicateurs à partir de la base de donnée type RFM
- Recherche d'un clustering via des essais (Kmean, Aggregatif, DBScan)
- Analyse des différents clusters
- Analyse de la stabilité dans le temps

Cleaning et feature engineering

Création d’un seul fichier de données

On part sur la base des indicateurs RFM :

- Récence : date du dernier achat
- Fréquence : fréquence des achats sur une période de référence donnée
- Montant : somme des achats cumulés sur cette période

On va étoffer ces features en extrayant un maximum de données à partir des tableaux disponibles et créer un tableau final avec une ligne par customer_unique_id.

Les features seront créées à partir de fonctions sur les tableaux

On nettoiera les données au fur et à mesure.

Exemples de fonction

```python
def heure(t):
    """ Retourne si le client achète plus souvent le matin, le soir, la nuit ou la journée"""

    df=orders[orders.order_purchase_timestamp<t][['order_id','customer_id','période_achat']]

    #On joint à droite, on a déjà éliminé l'order sans paiement
    final=pd.merge(df,customers[['customer_id','customer_unique_id']],on='customer_id',how='left')
    #On prends l'occurence la plus fréquente
    g=final.groupby('customer_unique_id').agg(periode=('période_achat',lambda x:x.value_counts().index[0]))
    return pd.DataFrame(g)
```

```python
def freight(t):
    """ calcule le pourcentage du prix du fret par order avant la date t
    renvois la moyenne de pourcentage sur l'ensemble des commandes"""

    df=orders[orders.order_purchase_timestamp<t][['order_id','customer_id']]

    df2=pd.merge(items[['order_id','price','freight_value']],df,on='order_id',how='left')
    price_order=df2.groupby(['order_id']).aggregate({'price':'sum','freight_value':'sum','customer_id':'first'}
    price_order['ratio_freight']=price_order['freight_value']/(price_order['price']+price_order['freight_value'

    cust_freight=pd.merge(price_order[['customer_id','ratio_freight']],customers[['customer_id','customer_uniqu
    return pd.DataFrame(cust_freight.groupby('customer_unique_id').aggregate({'ratio_freight':'mean'}))
```

Tableau : Customers

	customer_id	customer_unique_id	customer_zip_code_prefix	customer_city	customer_state
0	06b8999e2fba1a1fbc88172c00ba8bc7	861eff4711a542e4b93843c6dd7febb0	14409	franca	SP
1	18955e83d337fd6b2def6b18a428ac77	290c77bc529b7ac935b93aa66c333dc3	9790	sao bernardo do campo	SP
2	4e7b3e00288586ebd08712fdd0374a03	060e732b5b29e8181a18229c7b0b2b5e	1151	sao paulo	SP

- On récupère l'unique_id en index de notre tableau final data
- On conserve l'État
- On déduit les coordonnées GPS à partir du fichier de géolocalisation
- On rempli les données manquantes en faisant un 3-nn imputer à partir des codes postaux
- On fait la même chose pour les vendeurs

Tableau : Orders

	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	2017-10-04 19
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	2018-07-24 20:41:37	2018-07-26 03:24:27	2018-07-26 14

- On récupère la date du dernier achat
- On récupère la date du premier achat pour calculer la fréquence des achats
- On récupère le nombre de commandes
- On récupère la période la plus fréquente de la journée durant laquelle les achtas sont effectués (Matin, Journée, Soir ou Nuit)
- On va se servir de ce tableau pour les clés customer_id avec les autres tableaux

Tableau : Payments

	order_id	payment_sequential	payment_type	payment_installments	payment_value
85283	00010242fe8c5a6d1ba2dd792cb16214	1	credit_card	2	72.19
2499	00018f77f2f0320c557190d7a144bdd3	1	credit_card	3	259.83
12393	000229ec398224ef6ca0657da4fc703e	1	credit_card	5	216.87

On récupère :

- Le montant moyen de chaque paiement (installment)
- Le montant total des commandes
- Le type de paiement le plus utilisé

Tableau : Items

order_id	order_item_id	product_id	seller_id	shipping_limit_date	price	freight_value
0242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683c13e61	48436dade18ac8b2bce089ec2a041202	2017-09-19 09:45:35	58.9	13.29
18f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93d83a8f	dd7ddc04e1b6c2c614352b383efe2d36	2017-05-03 11:05:13	239.9	19.93

On récupère :

- La part moyenne du prix du fret sur le prix total
- Le nombre moyen d’items par commande
- Le vendeur favori de chaque client (en montant total)

Tableau : Products

	product_id	product_category_name	product_name_lenght	product_description_lenght	product_photos_qty	product_weight_g	proc
0	1e9e8ef04dbcff4541ed26657ea517e5	perfumaria	40.0	287.0	1.0	225.0	
1	3aa071139cb16b67ca9e5dea641aaa2f	artes	44.0	276.0	1.0	1000.0	
2	96bd76ec8810374ed1b65e291975717f	esporte_lazer	46.0	250.0	1.0	154.0	

On récupère :

- Le nombre de catégories différentes achetées
- La catégorie favorite (en montant total)

Tableau : Reviews

- On évite les duplicates en ne gardant que la dernière review de chaque commande.

	review_id	order_id	review_score	review_comment_title	review_comment_message	review_creation_dat
0	7bc2406110b926393aa56f80a40eba40	73fc7af87114b39712e6da79b0a377eb	4	NaN	NaN	2018-01-18 00:00:0
1	80e641a11e56f04c1ad469d5645fdfde	a548910a1c6147796b98fdf73dbeba33	5	NaN	NaN	2018-03-10 00:00:0

- On récupère le score moyen donné sur l’ensemble des commandes

Tableau : Sellers

	seller_id	seller_zip_code_prefix	seller_city	seller_state	Latitude	Longitude
0	3442f8959a84dea7ee197c632cb2df15	13023	campinas	SP	-22.893848	-47.061337
1	d1b65fc7debc3361ea86b5f14c68d2e2	13844	mogi guacu	SP	-22.383437	-46.947927

Graces aux coordonnées GPS :

- On calcule le poids.distance moyen effectué par chaque commande

Tableau final :

customer_unique_id	total_number_of_orders	state	Latitude	Longitude	favorite_period	total_payment_value	mean_installment
0000366f3b9a7992bf8c76cfdf3221e2	1.0	SP	-23.340235	-46.830140	morning	141.90	17.7375
0000b849f77a49e4a4ce2b2a4ca5be3f	1.0	SP	-23.559115	-46.787626	morning	27.19	27.1900

favorite_payment_type	mean_ratio_freight	items_per_order	favorite_seller_id	nb_of_category	favorite_category	mean_review_score
credit_card	0.084567	1.0	da8622b14eb17ae2831f4ac5b9dab84a	1	cama_mesa_banho	5.0
credit_card	0.304892	1.0	138dbe45fc62f1e244378131a6801526	1	beleza_saude	4.0

ecology	days_since_last_order	procurement_fréquency
1569.867304	937	0.001067
80.771515	940	0.001064

Présentation des pistes de modélisation

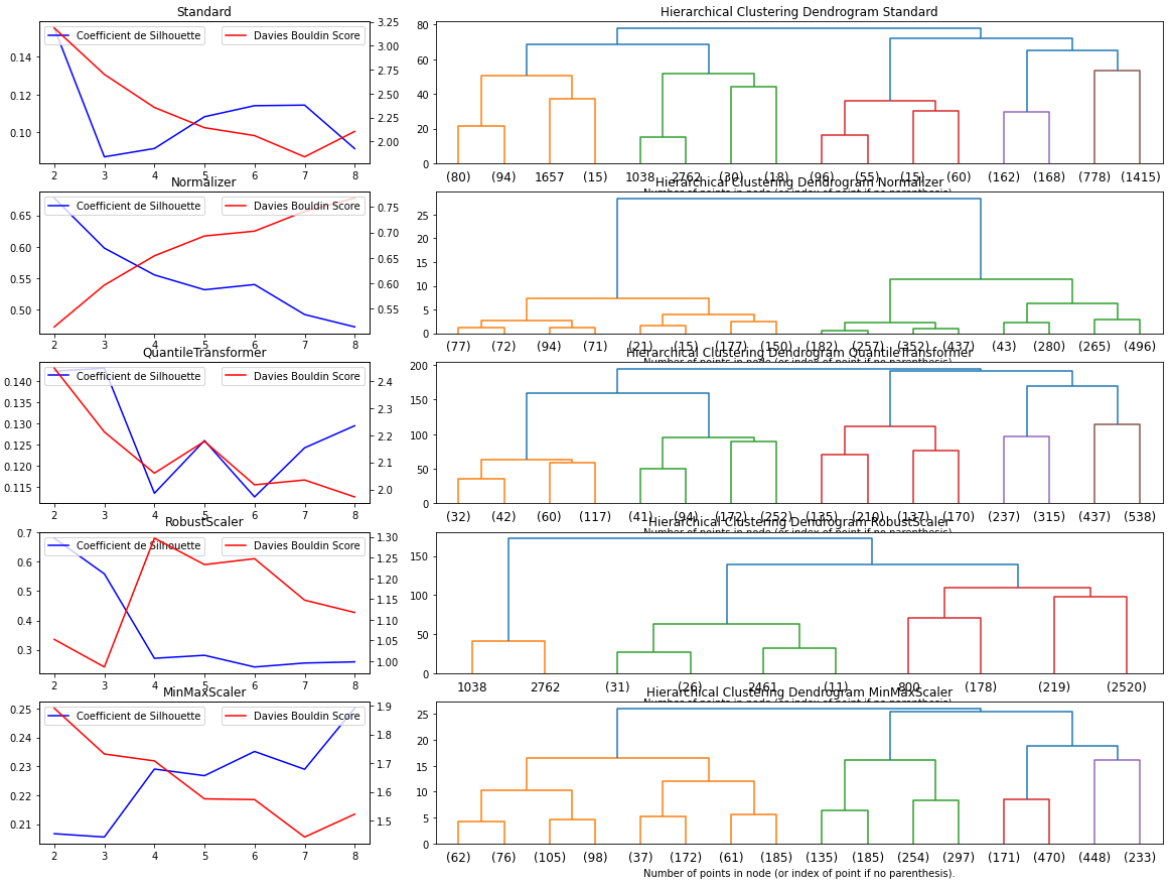
Choix de la normalisation

On prend un sous échantillon des clients ayant effectués plusieurs commandes.

On code toutes nos variables catégorielles dans un premier temps en leur affectant la proportion de la catégorie dans l'échantillon.

On regarde sur des Kmeans et des cluster hiérarchiques les résultats.

Le Normalizer nous donne des résultats clairement supérieurs sur les scores de Silhoutte et Davies Bouldin. On part la dessus

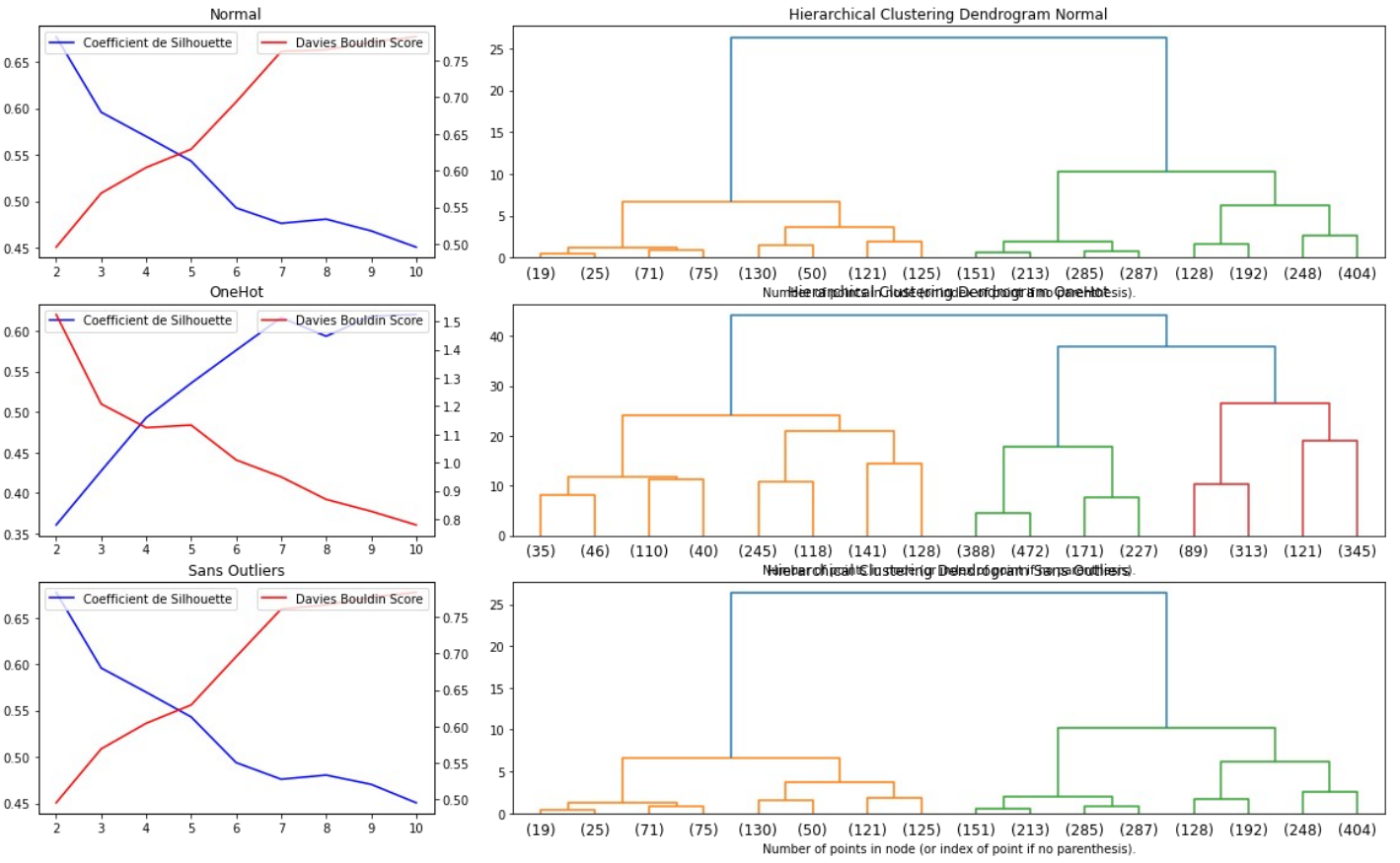


Codage et outliers

On reste avec notre codage catégoriel.

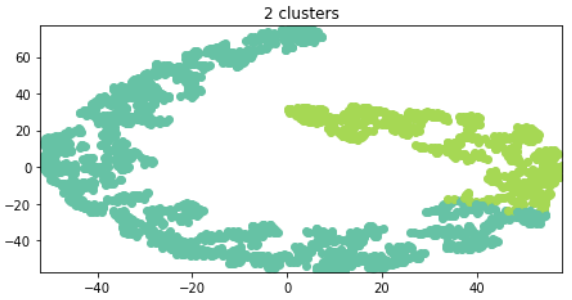
On conserve les outliers.

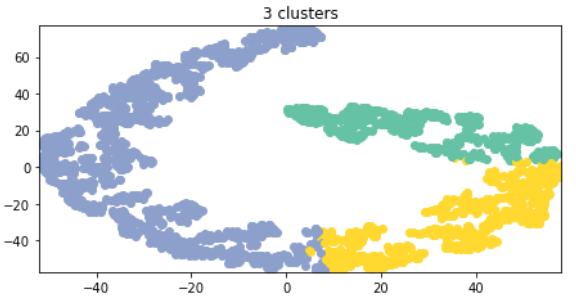
On va regarder ce que cela nous donne pour 2 à 6 clusters via isomap et t-sne.

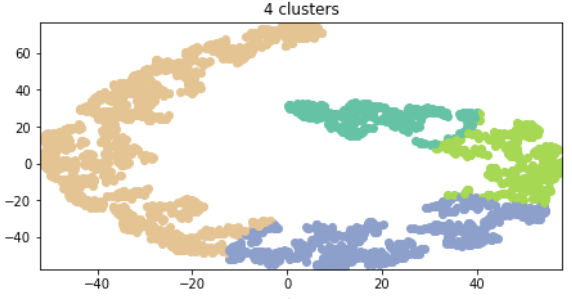


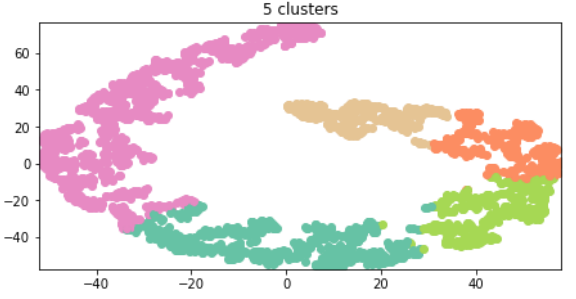
Kmean et T-SNE

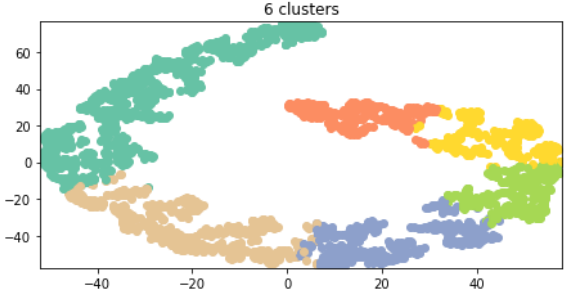
La répartition en 5 clusters semble la plus pertinente





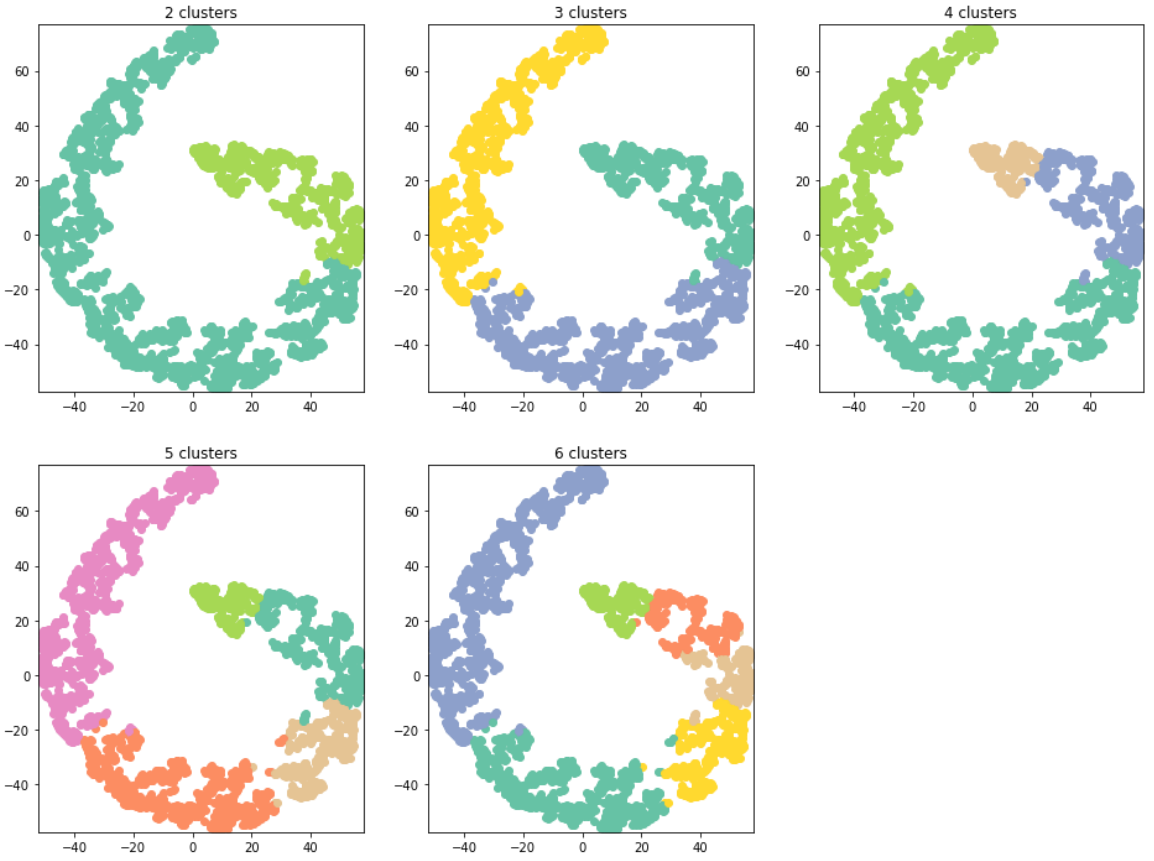




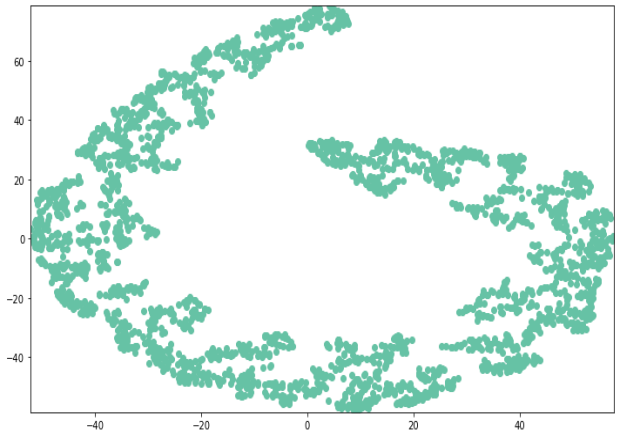


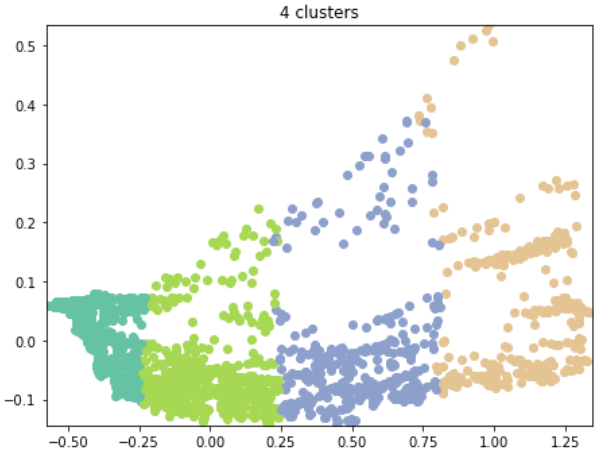
Hiérarchique et TSNE

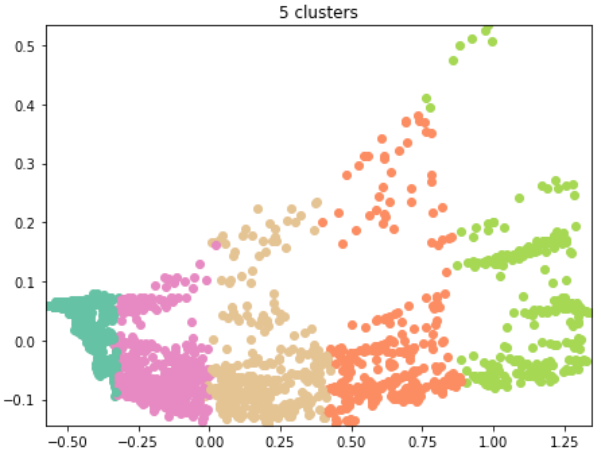
On retrouve à peu près les mêmes clusters



DBSCAN et Isomap





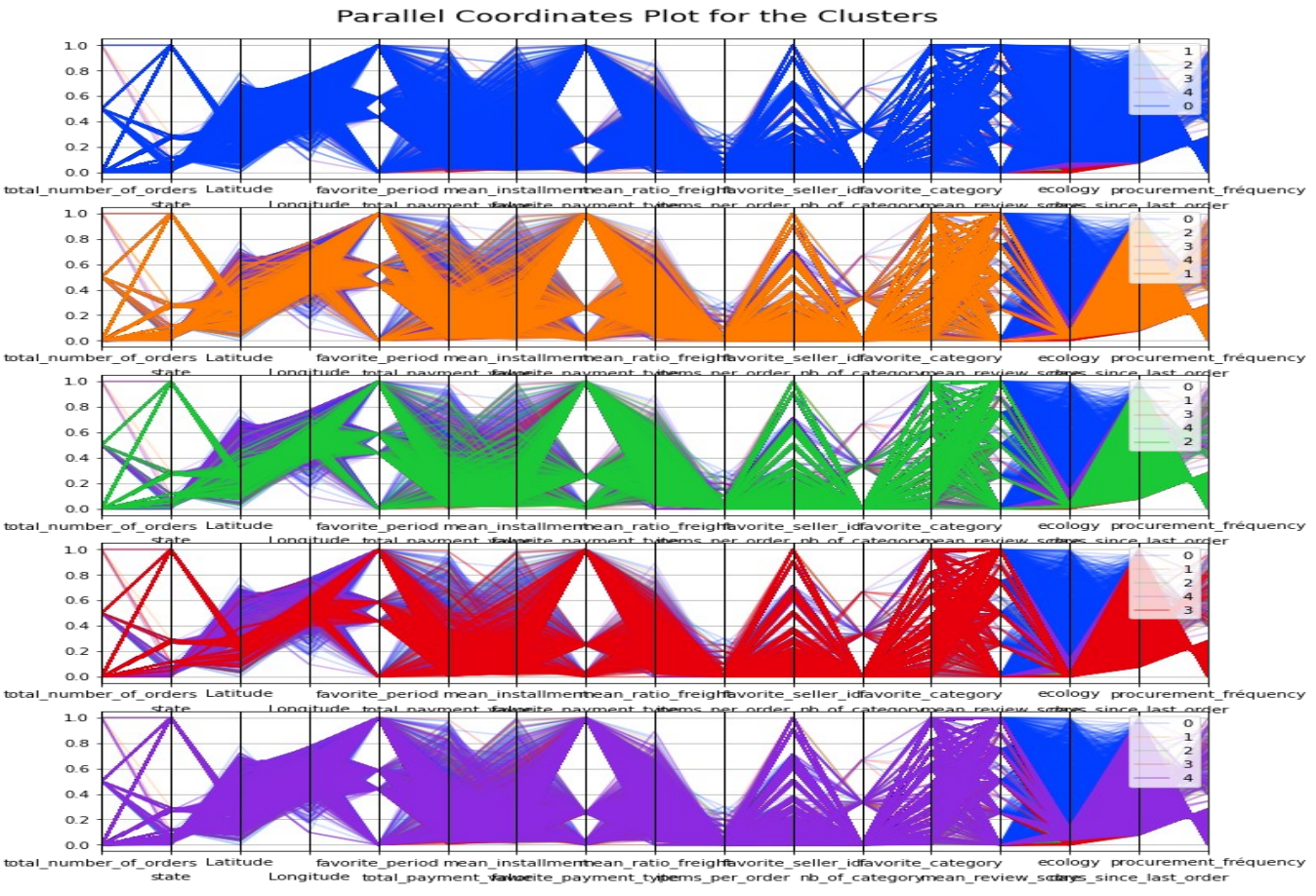


5 clusters

A première vue, en dehors de « écology » rien de très discriminant.
Les variances des moyennes interclusters et les variances au sein des clusters :

```
#regardons les variances des moyennes interclusters:
a=X_clustered.groupby('cluster').mean().mean()
b=X_clustered.groupby('cluster').mean().std()
b/a
```

```
total_number_of_orders     0.194574
state                      0.339770
Latitude                   0.112697
Longitude                  0.012200
favorite_period            0.004242
total_payment_value        0.297118
mean_installment           0.171667
favorite_payment_type      0.012725
mean_ratio_freight         0.080198
items_per_order            0.395445
favorite_seller_id         0.053078
nb_of_category             0.385592
favorite_category          0.035214
mean_review_score          0.023064
ecology                    1.491376
days_since_last_order      0.023621
procurement_fréquency      0.018687
dtype: float64
```



Nettoyage de variables

- On supprime Longitude, Type de Paiement, Période d’achat, review_score, temps depuis le dernier paiement et fréquence d’achat.

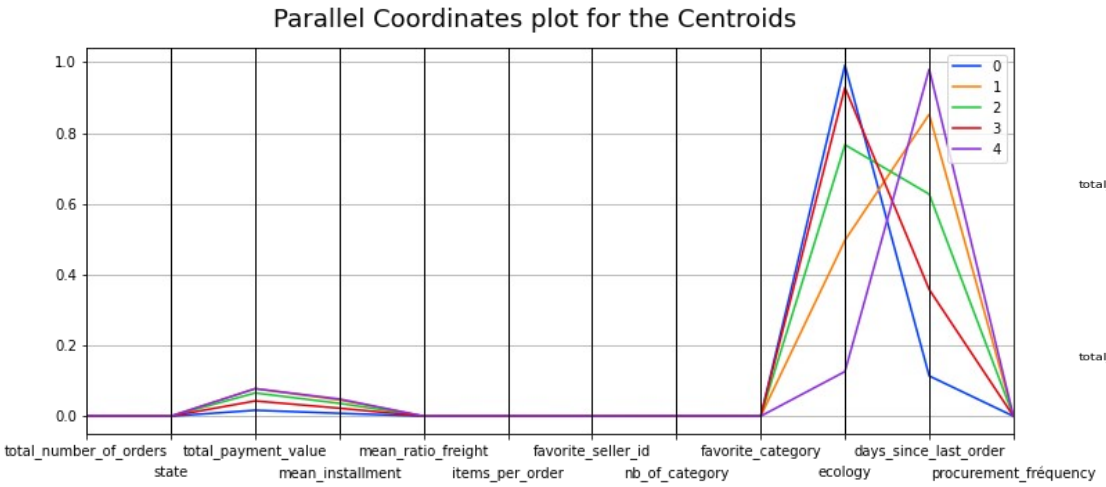
  On vérifie que les variation intracluster sont similaires également :

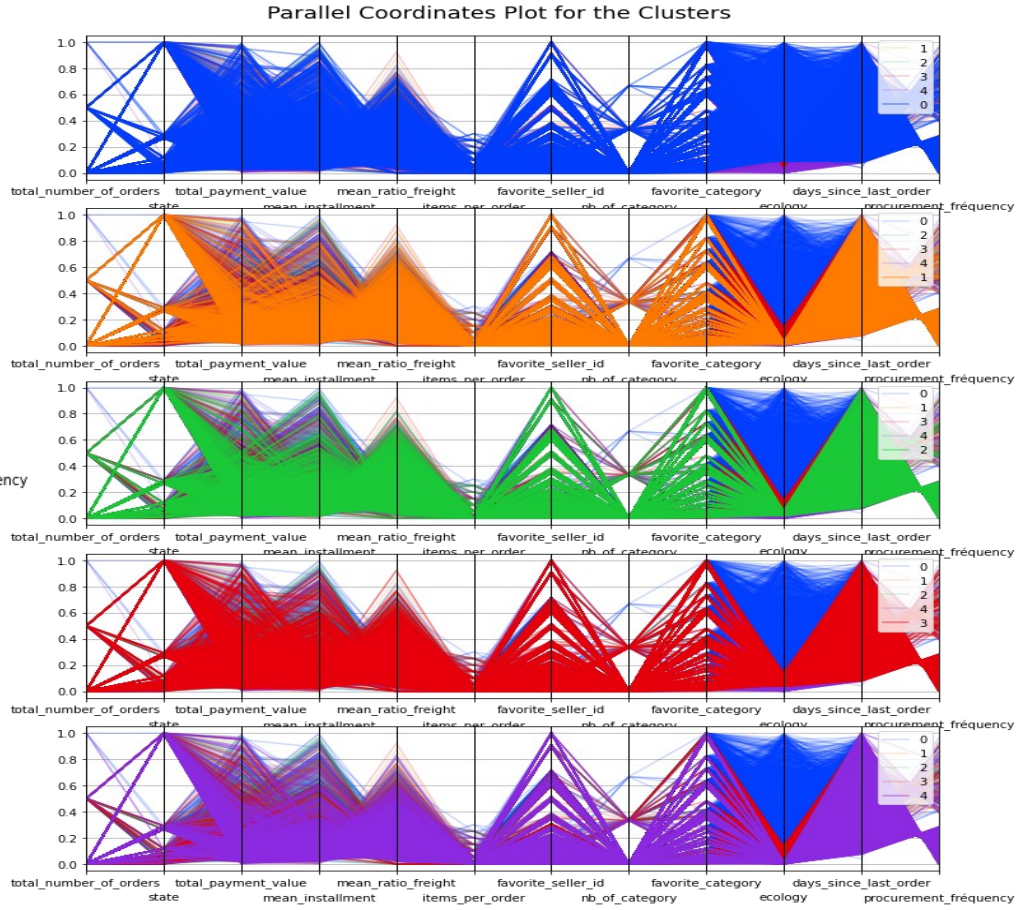
- On améliore très légèrement nos scores :

  Le coefficient de silhouette passe de 0.5511 à 0.5516

  Le coefficient de Davies-Bouldin de 0.6001 à 0.5992

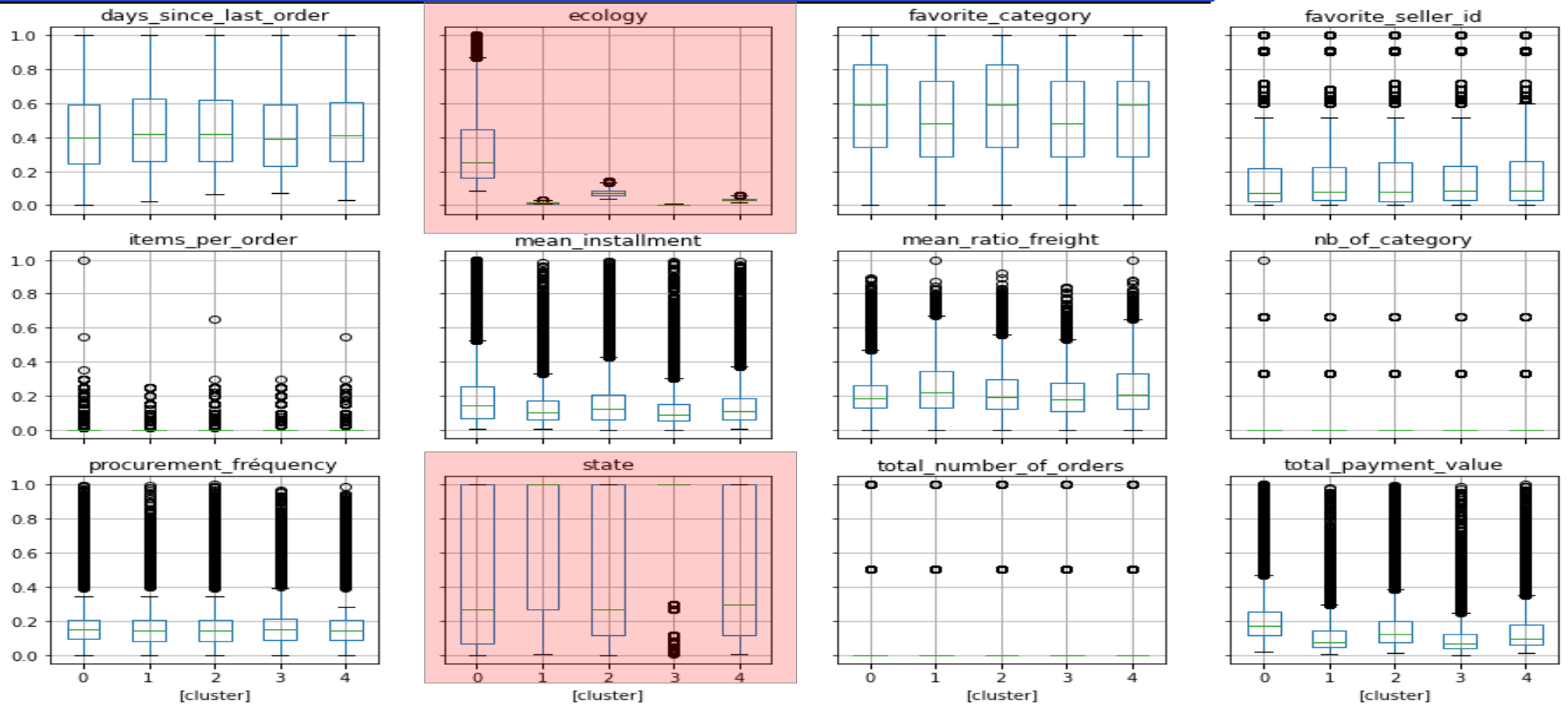
Premières infos sur les clusters



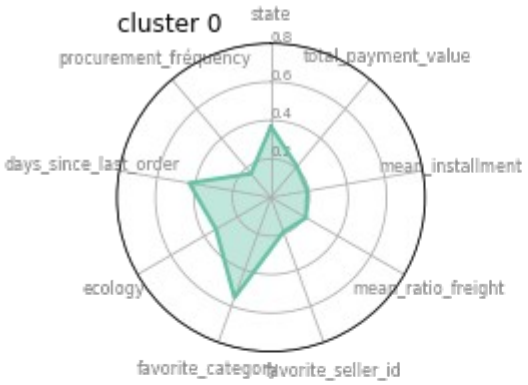


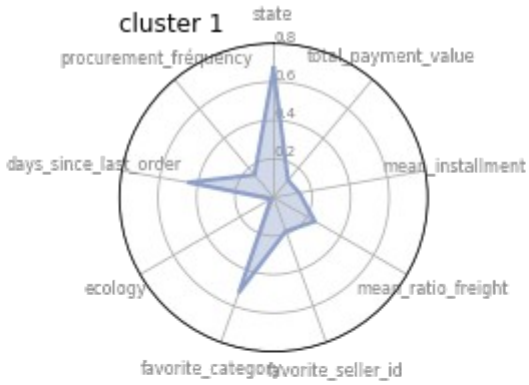
Ce qui semble différencier les cluster est :
Le paiement total
La moyenne des installments
L’ « écologie »
Le nombre de jours depuis le dernier achat

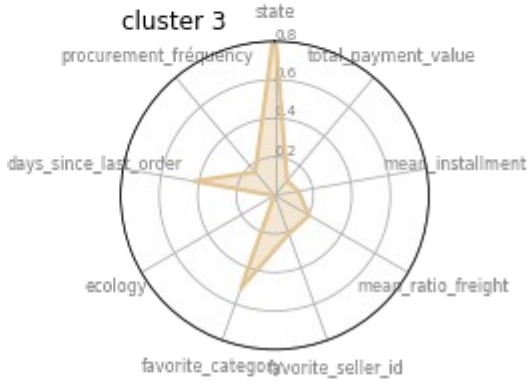
Diagrammes en boites (MinMax)

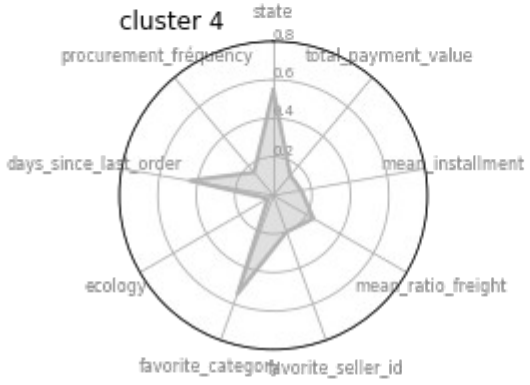


Moyennes de chaque cluster

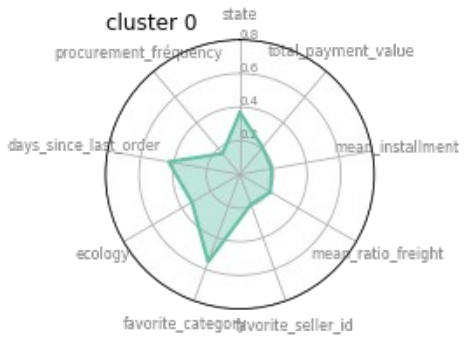




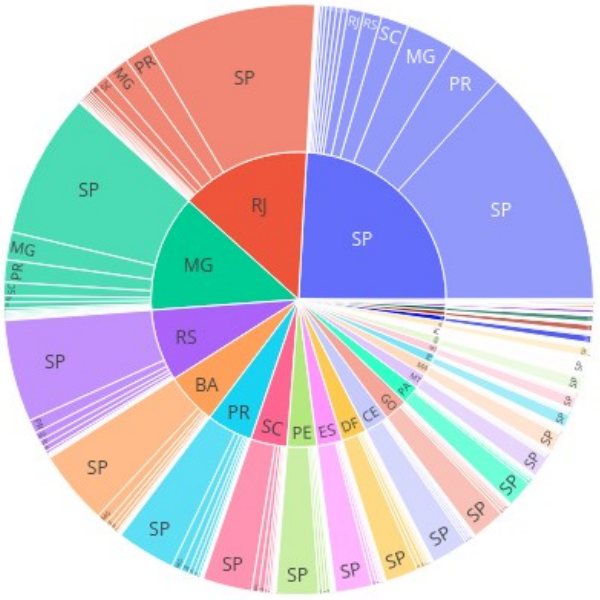




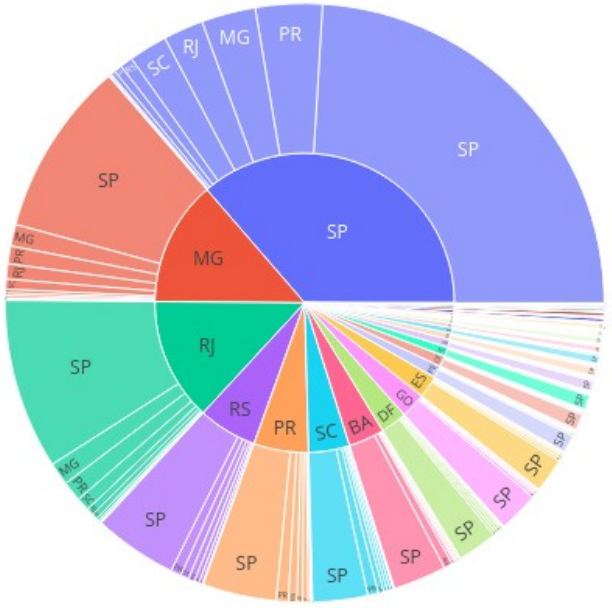
Cluster 0 (39915-42 %): Ceux qui ont des attentes précises

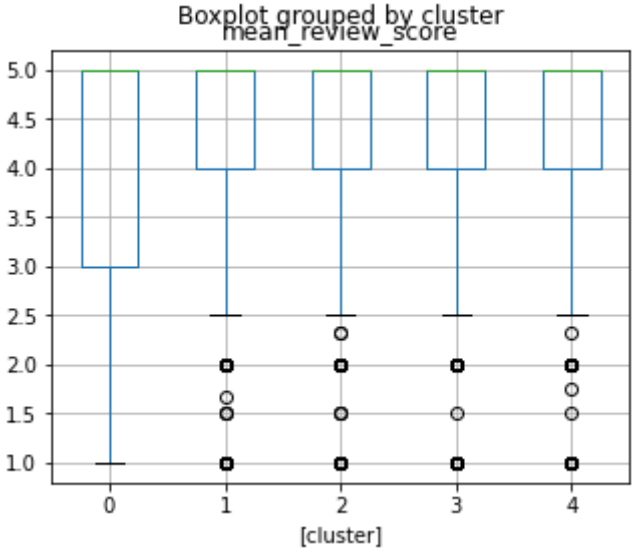


Régions Acheteurs et Vendeurs: Cluster0



Régions Acheteurs et Vendeurs: Base de donnée





Plus exigeants, achètent plutôt ce qu'ils ne trouvent pas chez eux, le panier moyen est plus important

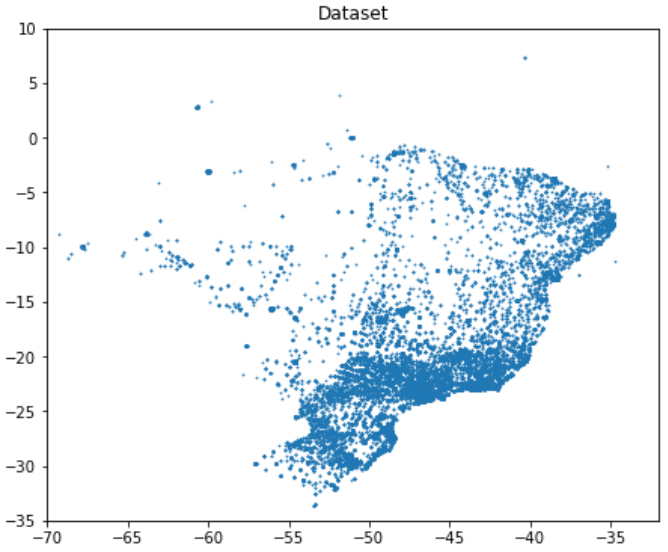
Conseil : Proposer des articles plus spécifiques

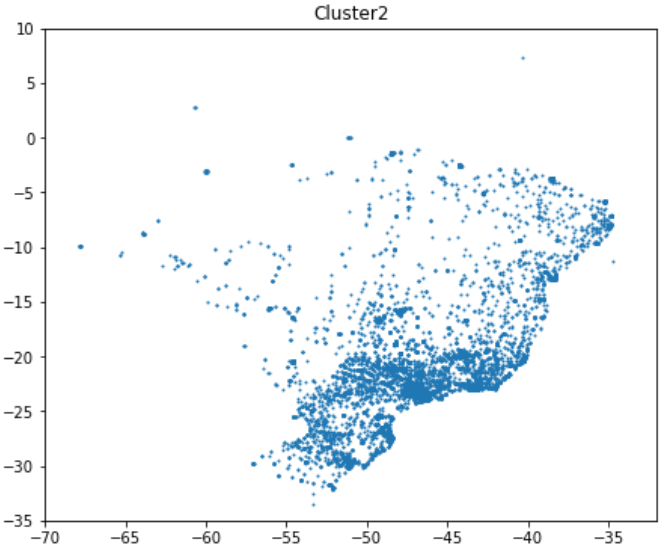
Cluster 1 : Les clients faciles

- Pas de gros acheteurs mais :
  - Plutôt satisfaits
  - Pas regardant sur les prix des transports (Coûts élevés malgré des petites distances)
- **Conseil :** Fidéliser avec des bons d'achats

cluster	total_payment_value	mean_ratio_freight	items_per_order	nb_of_category	mean_review_score	ecology
0	232.121899	0.197791	1.206433	1.034999	3.995606	29501.940444
1	107.927522	0.234655	1.074289	1.018172	4.178968	634.190733
2	143.848817	0.212749	1.114381	1.026546	4.110248	2931.001499
3	91.009297	0.196205	1.080408	1.012256	4.218071	138.549911
4	123.561592	0.227009	1.082346	1.019820	4.116886	1338.722676
All	166.286761	0.208494	1.139081	1.026148	4.086633	13164.871669

Cluster 2 : Ceux qui cherchent les bonnes affaires





Même distribution géographique

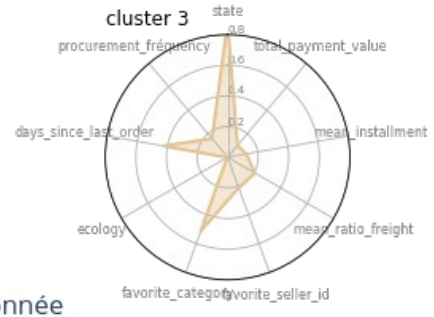
Des poids distance importants, avec plus d'items par order et plus de catégories différentes par order, donc potentiellement des coûts de transports plus importants mais ce n'est pas le cas.

Conseil : cible pour promos

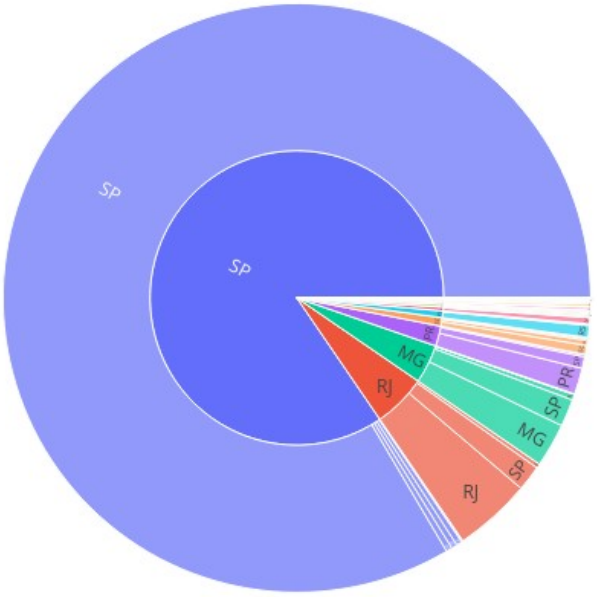
cluster	total_payment_value	mean_ratio_freight	items_per_order	nb_of_category	mean_review_score	ecology
0	232.121899	0.197791	1.206433	1.034999	3.995606	29501.940444
1	107.927522	0.234655	1.074289	1.018172	4.178968	634.190733
2	143.848817	0.212749	1.114381	1.026546	4.110248	2931.001499
3	91.009297	0.196205	1.080408	1.012256	4.218071	138.549911
4	123.561592	0.227009	1.082346	1.019820	4.116886	1338.722676

cluster	total_payment_value	mean_installment	mean_ratio_freight	ecology
0	153.610	64.8900	0.178994	12728.225759
1	66.830	42.6900	0.207478	625.477631
2	99.900	53.0100	0.183031	2815.756854
3	57.600	37.8300	0.173157	112.816765
4	81.655	46.2825	0.197133	1299.537659

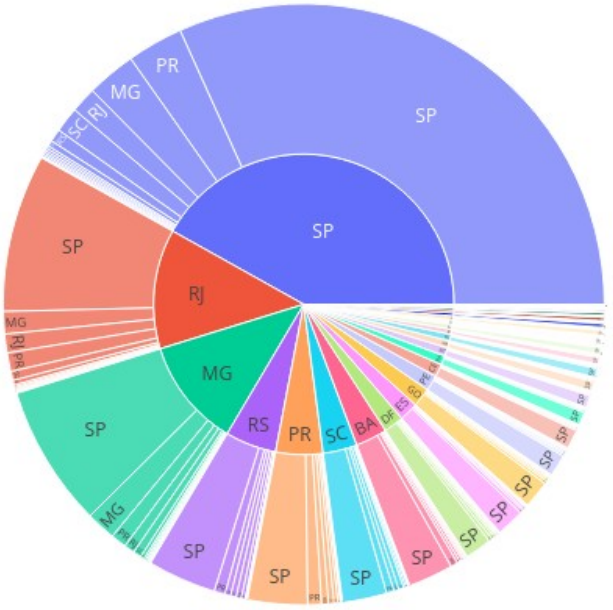
Cluster 3 : Les identitaires (13871-13%)



Régions Acheteurs et Vendeurs Cluster3



Régions Acheteurs et Vendeurs: Base de donnée



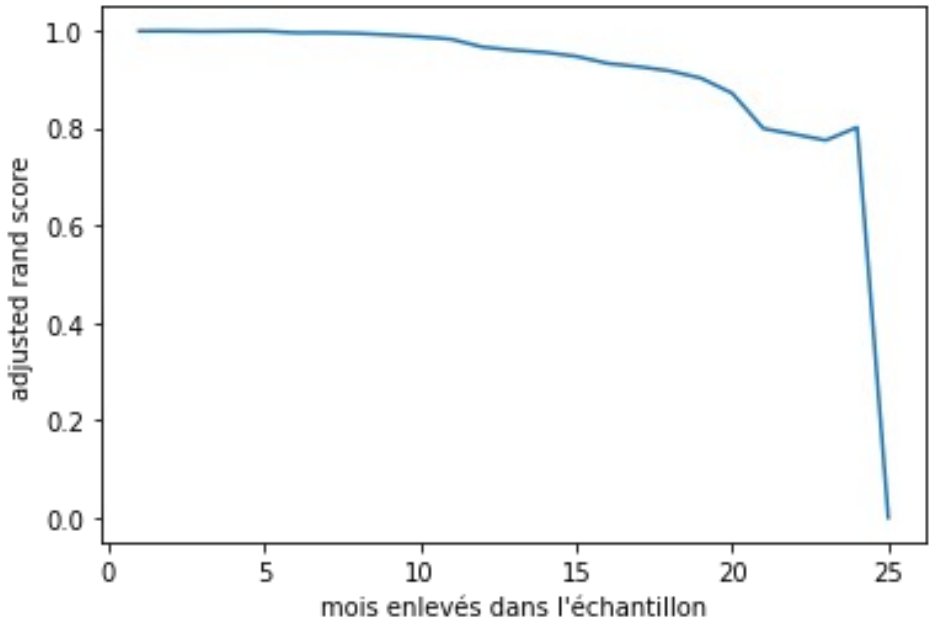
N’achètent quasi exclusivement que dans leurs région/ville ou les régions limitrophes

Conseil : Proposer des produits de vendeurs à proximité

Cluster 4 : Les Autres...

- Je n’ai pas réussi à trouver de caractéristiques sur ce groupe

Stabilité des clusters dans le temps



Ainsi le clustering reste relativement stable sur 18 à 20 mois, on peut donc proposer une maintenance tous les 18-20 mois.