Projet n°4 : Segmentez les clients d'un site de e-commerce

01

Introduction

Présentation du problème, présentation des données

02

Analyses

Analyses des caractéristiques et comportements des clients

03

Segmentations

KMeans, DBSCAN

04

Conclusion

Types de clients, contrat de maintenance

01 Introduction

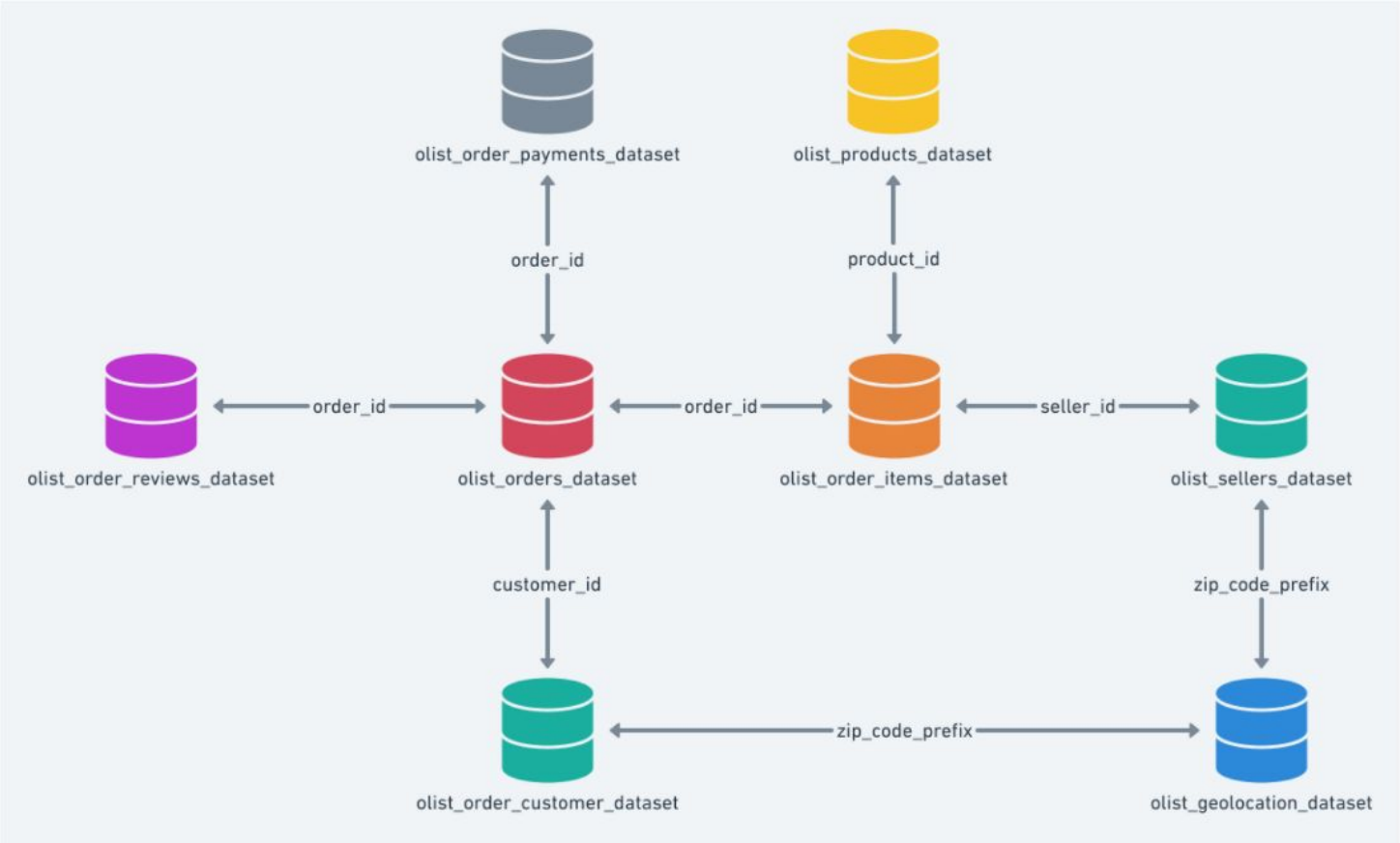
Présentation de la problématique

- Olist est une entreprise brésilienne qui propose une solution aux petites entreprises qui souhaitent vendre leurs produits en ligne. Elle leur permet d'être plus visibles et donc à toucher une clientèle plus large.

- Olist a besoin a besoin d'une meilleure vision des clients (i.e ceux qui achètent les produits en ligne).

- Notre but : réaliser une segmentation client, exploitable et facile d'utilisation pour l'équipe marketing

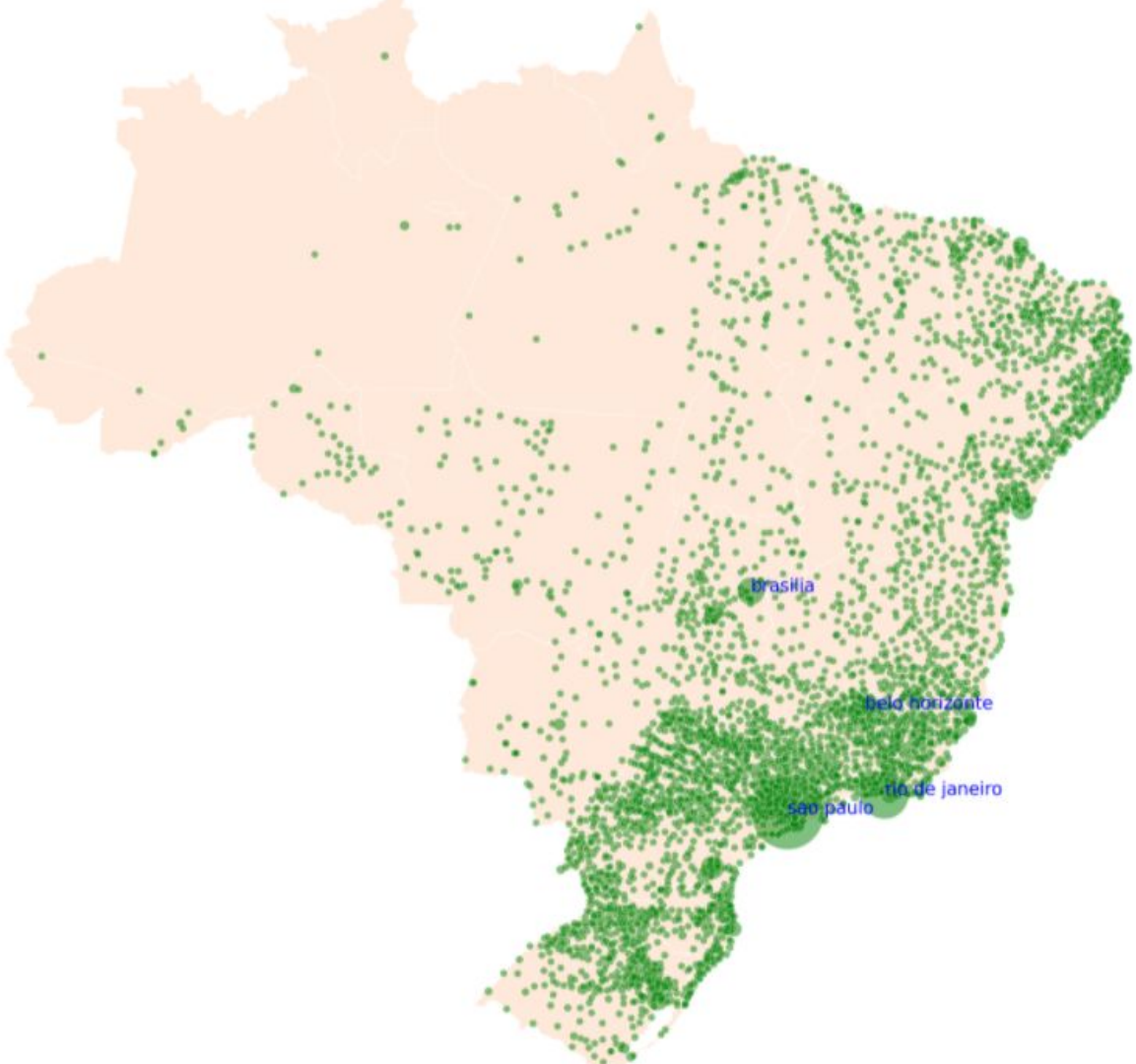
Présentation des données



02 Analyses

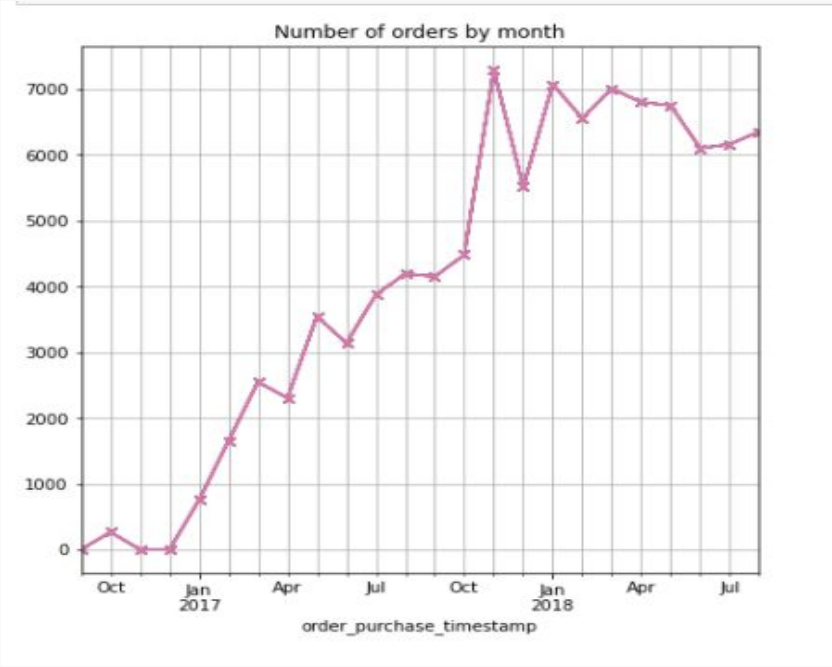
Où sont situés les clients ?

- Environ 96 000 clients
- Définition de la variable ‘city density’
- Davantage de clients dans les grandes villes que sur la côte
- Peu de clients dans les terres vers le nord ouest

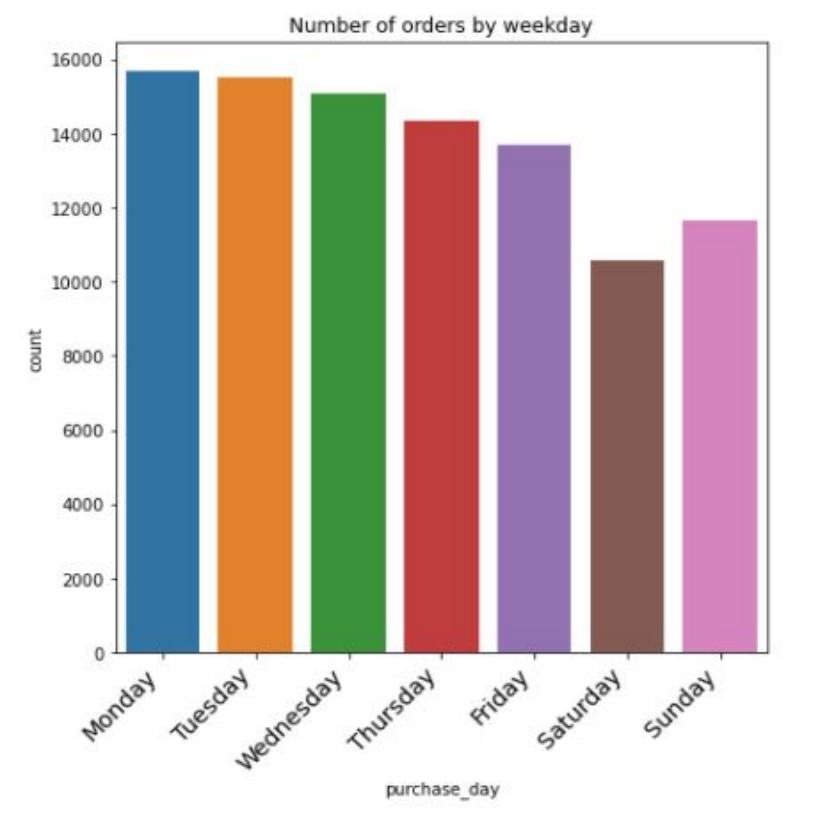


Nombre de commandes et évolution au fil du temps

Environ 99 000 commande: 96% des clients ont effectué une seule commande.

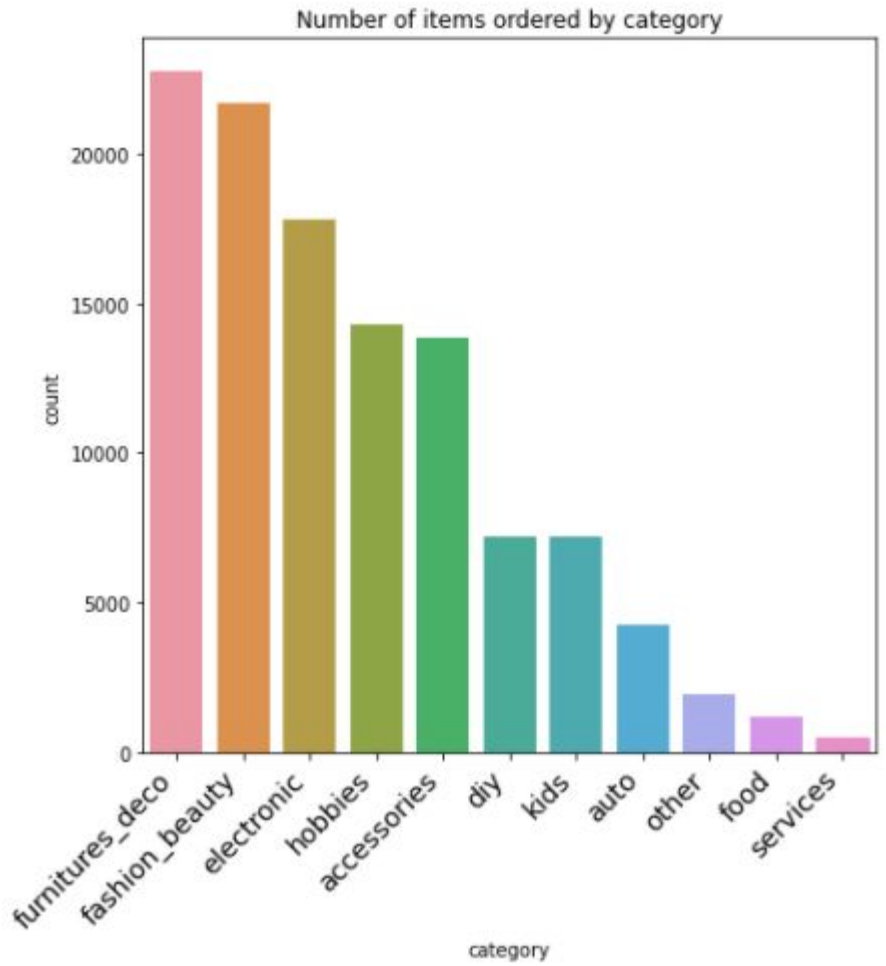


Nombre de commandes et évolution au fil du temps

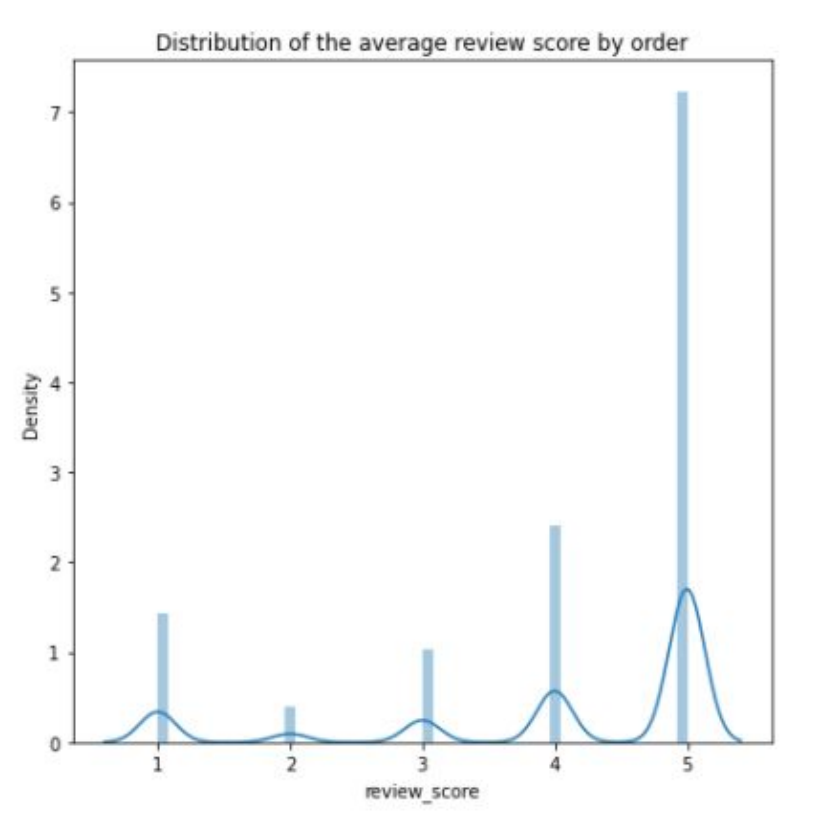


En moyenne on a davantage de commandes en semaine que le week end

Quels types de produits les clients achètent-ils ?



Satisfaction des clients



Les produits ont dans l'ensemble des notes très satisfaisantes.

Satisfaction des clients

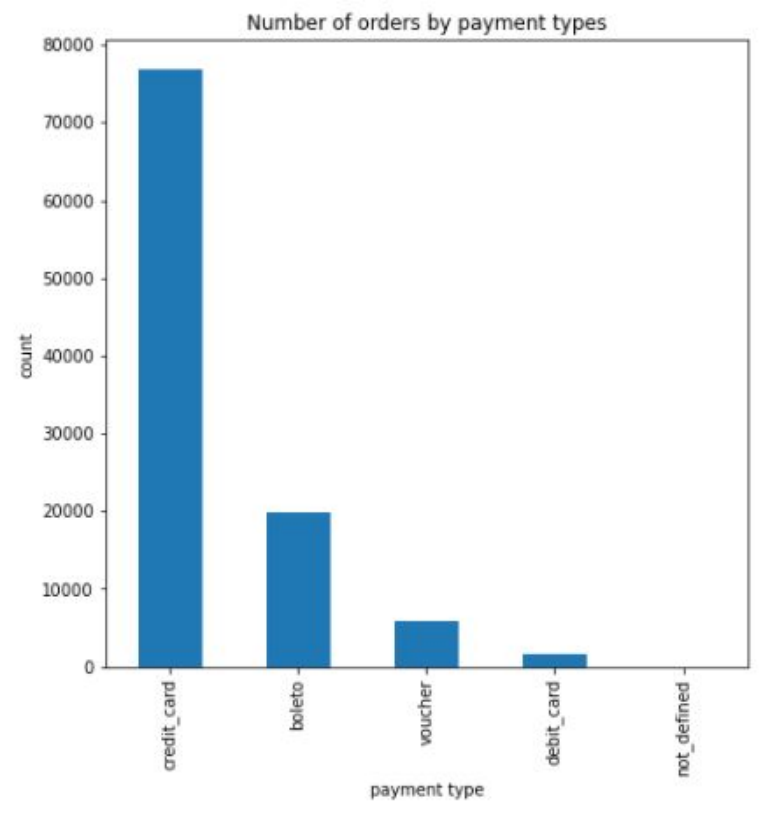
Mots qui reviennent le + souvent dans les commentaires des clients insatifaits (<⅖): ‘produit’ et ‘reçu”

```
word_freq(customers_unsatisfied,'review_comment_message')
```

```
[('produto', 4122),
 ('recebi', 2564),
 ('comprei', 1403),
 ('para', 1293),
 ('ainda', 1075),
 ('veio', 1073),
 ('entrega', 802),
 ('estou', 750),
 ('entregue', 746),
 ('chegou', 733),
 ('mais', 589),
 ('muito', 584),
 ('compra', 573),
 ('agora', 555),
 ('prazo', 555),
 ('como', 516),
 ('minha', 506),
 ('pedido', 484),
 ('loja', 460),
 ('apenas', 432)]
```

(fonction word_freq en annexe)

Paiements




```
: payment_types['nb_payments'].describe(np.arange(0,1,0.1))

count    99440.000000
mean         1.044680
std          0.381209
min          0.000000
0%           0.000000
10%          1.000000
20%          1.000000
30%          1.000000
40%          1.000000
50%          1.000000
60%          1.000000
70%          1.000000
80%          1.000000
90%          1.000000
max         29.000000
Name: nb_payments, dtype: float64
```

Panier moyen

```
payments['payment_value'].describe()
```

```
count    103886.000000
mean        154.100380
std         217.494064
min           0.000000
25%          56.790000
50%         100.000000
75%         171.837500
max       13664.080000
Name: payment_value, dtype: float64
```

03 Segmentation

Agrégation des données

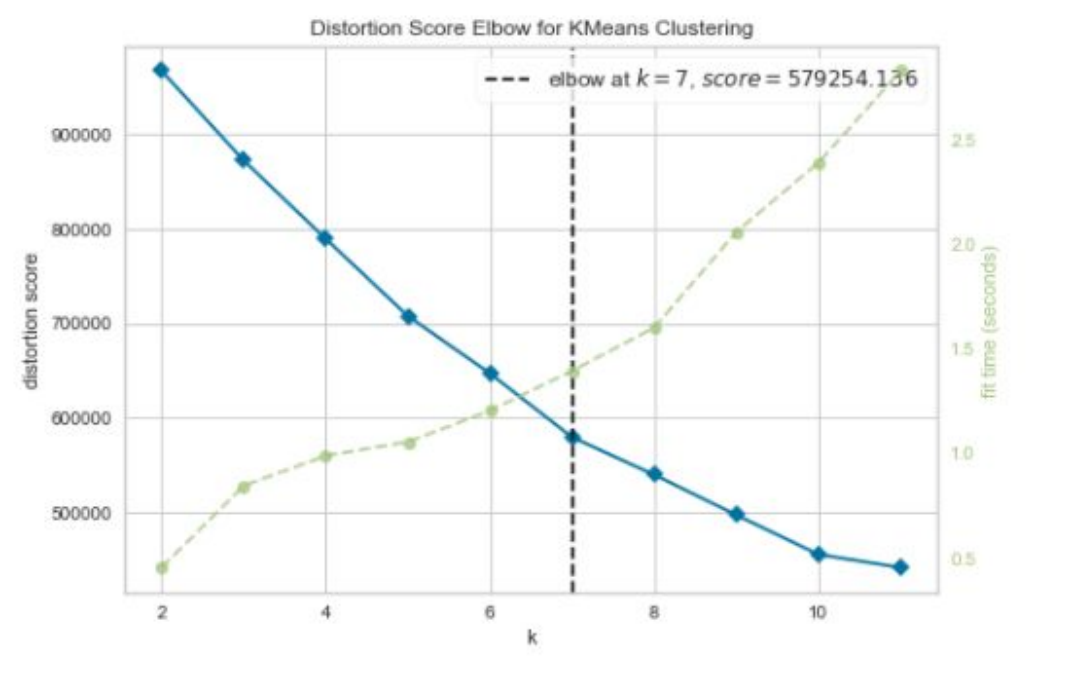
On créée un dataset indexé par id client unique avec notamment :

- Les variables RFM : le nombre de jour depuis le dernier achat, le nombre de commandes et le montant moyen des commandes
- Le nombre de paiements moyen, le nombre d'utilisation des différents types de paiements moyen par commande
- Le nombre d'articles moyen, les dimensions moyennes par commande
- Le score moyen
- La densité de la ville du client

annexe 2

1ère segmentation KMeans

Choix du nombre de clusters K



1ère segmentation KMeans

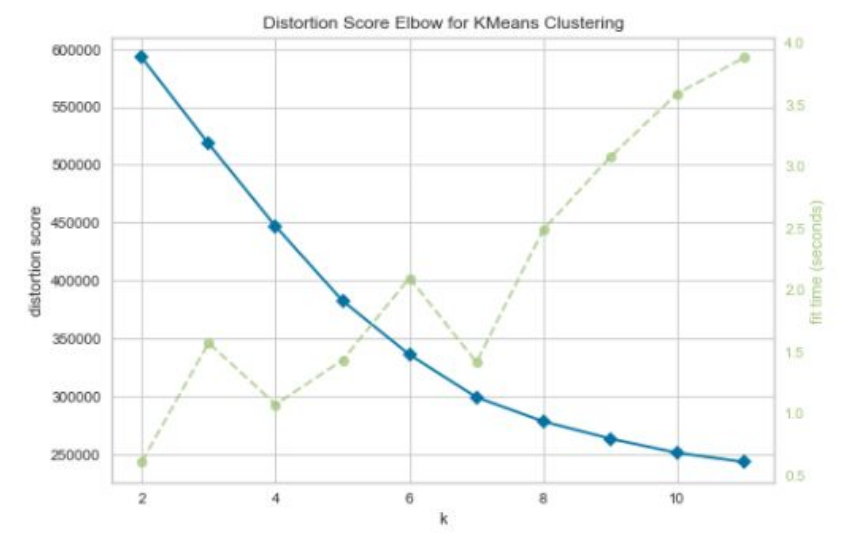
K=7

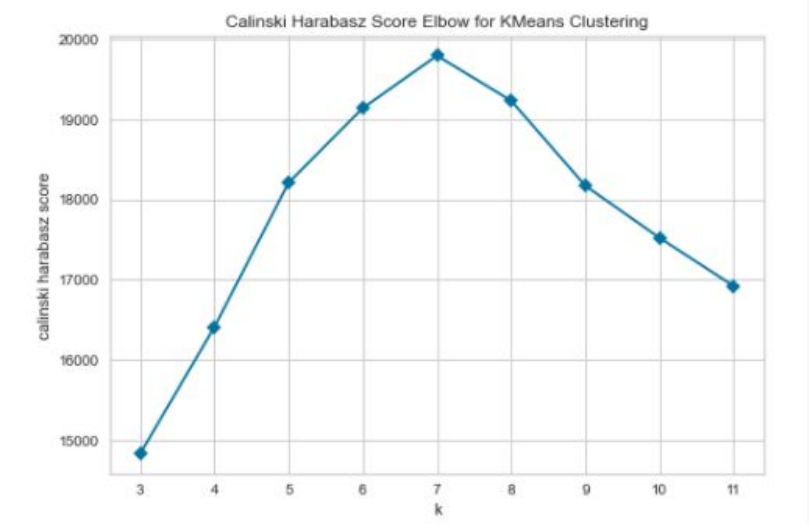
```
4    48414
2    17738
0    10847
3    10666
1     2477
5     1431
6      515
Name: num_cluster, dtype: int64
```

	payme…e_credit_card	payment_type_boleto	payment_ty…it_card	payment_ty…cher
0	0.982000	0.002000	0.000000	0.043000
1	0.857000	0.137000	0.001000	0.036000
2	0.000000	0.999000	0.000000	0.001000
3	0.967000	0.021000	0.000000	0.044000
4	0.986000	0.001000	0.000000	0.044000
5	0.004000	0.006000	0.990000	0.000000
6	0.642000	0.001000	0.000000	3.877000

	nb_of_orders	nb_…nts	nb_days_since_last_order	delivery_nb_of_days	payment_value	nb_items	review_score	city_density	dimensions
0	1.043000	1.031000	217.503000	6.744000	130.069000	1.127000	4.344000	16.160000	20173.691000
1	1.027000	1.031000	272.045000	13.052000	273.287000	1.102000	4.186000	2.800000	242816.320000
2	1.028000	1.000000	245.644000	12.483000	129.448000	1.147000	4.193000	3.030000	20125.807000
3	1.026000	1.032000	240.248000	23.050000	182.266000	1.369000	1.703000	1.706000	21957.553000
4	1.036000	1.031000	237.026000	10.648000	145.678000	1.086000	4.636000	1.009000	19653.744000
5	1.031000	1.001000	166.166000	10.280000	124.775000	1.113000	4.246000	4.009000	22428.816000
6	1.041000	4.520000	264.922000	12.249000	130.287000	1.117000	4.048000	3.199000	31688.157000

2e segmentation KMeans



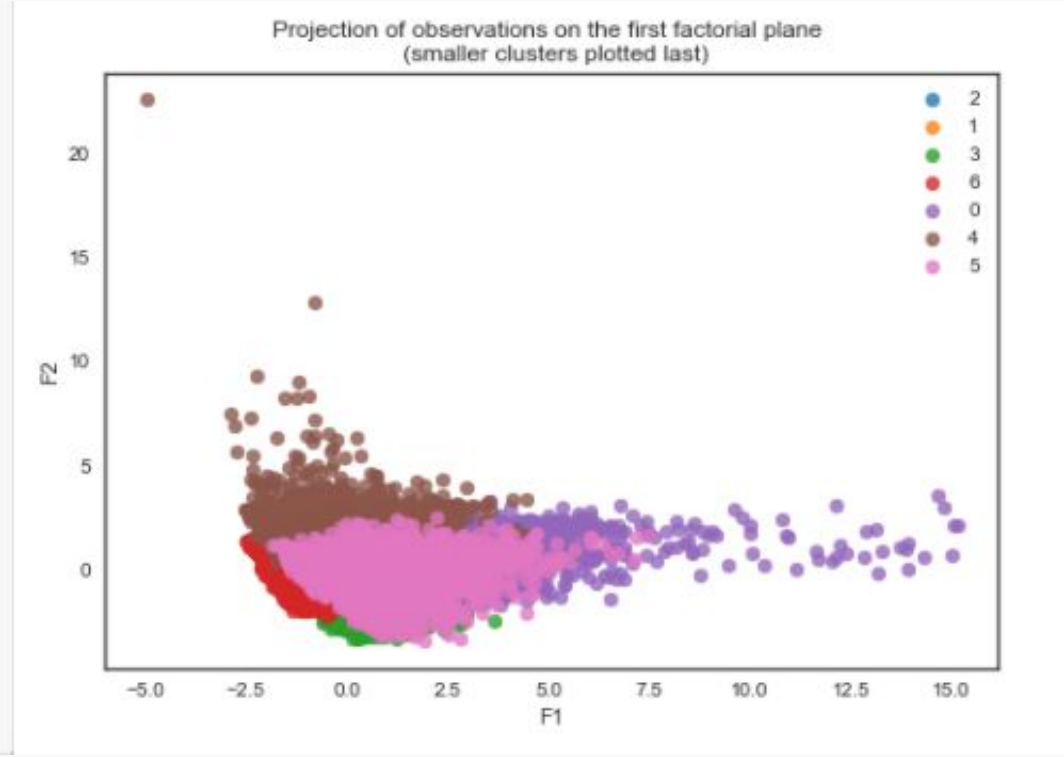


2e segmentation KMeans

K=7

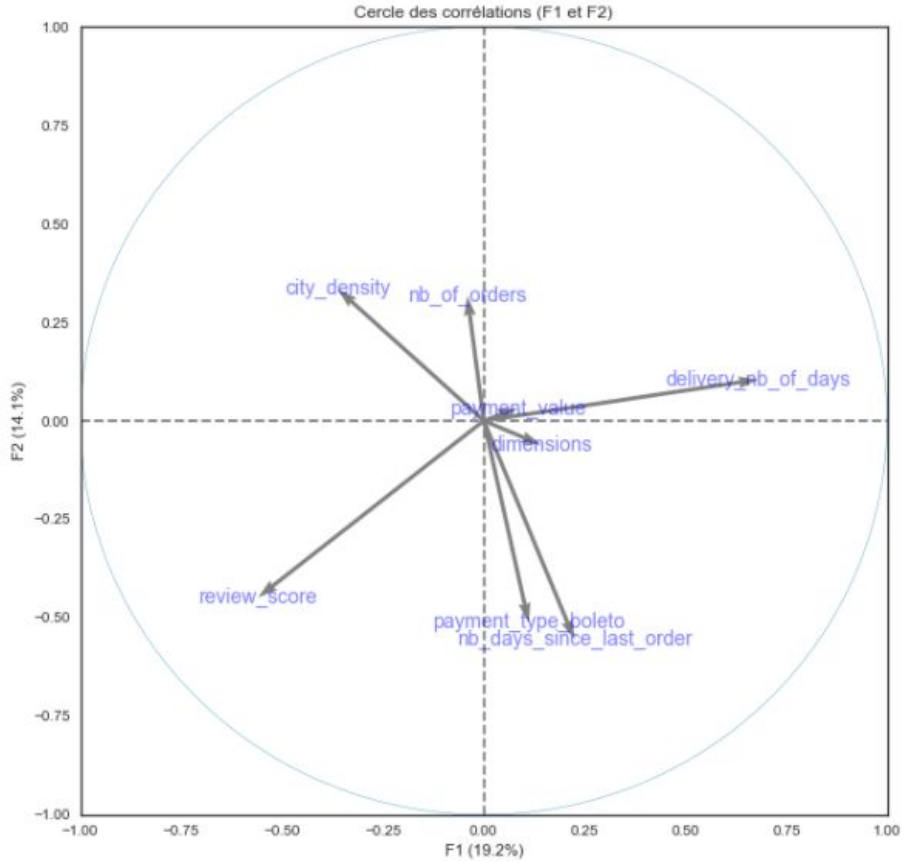
	nb_of_orders	payment_type_boleto	nb_days_since_last_order	delivery_nb_of_days	payment_value	review_score	city_density	dimensions
0	1.000000	0.123000	233.312000	24.258000	172.986000	1.503000	1.775000	21865.765000
1	1.000000	0.000000	388.507000	11.546000	148.902000	4.575000	1.051000	21561.958000
2	1.000000	0.000000	121.623000	10.029000	146.241000	4.619000	0.984000	18564.841000
3	1.000000	1.000000	249.053000	12.305000	128.871000	4.420000	0.819000	19229.825000
4	2.115000	0.189000	218.888000	11.826000	143.206000	4.205000	3.793000	26449.650000
5	1.006000	0.166000	269.385000	13.090000	272.616000	4.171000	2.861000	247042.832000
6	1.000000	0.182000	219.526000	6.881000	128.062000	4.333000	16.171000	20058.177000

2e segmentation KMeans avec K=7



On a 33 % du taux de variance expliqué sur le 1er plan factoriel

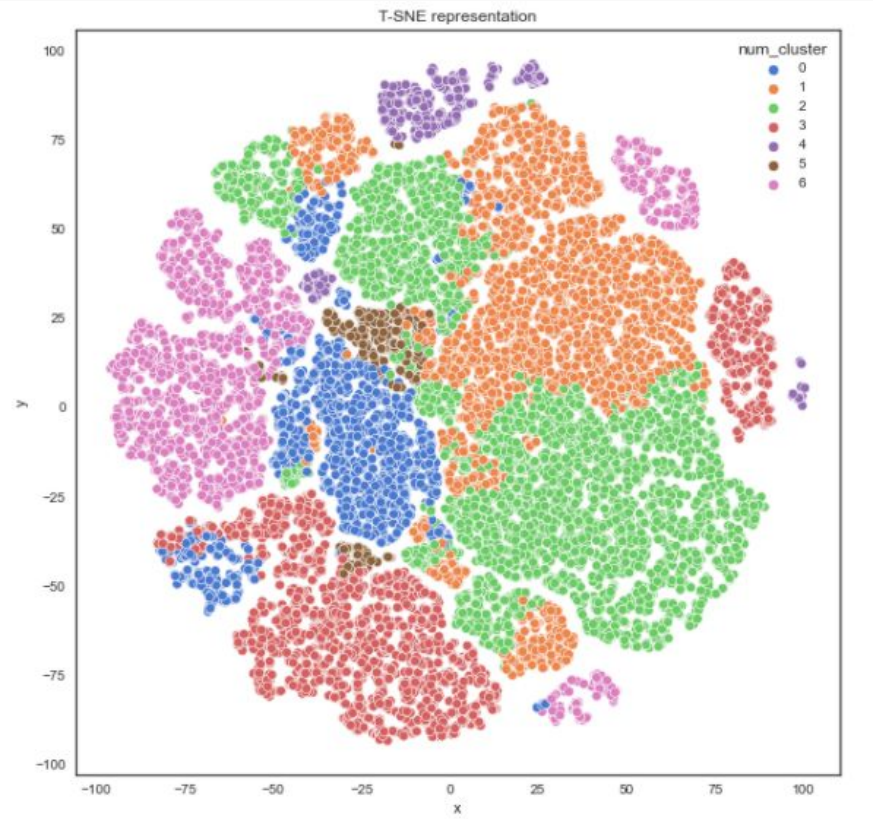
2e segmentation KMeans avec K=7



PC1: efficacité du service auprès du client

PC2: Fidélisation du client

2e segmentation KMeans avec K=7



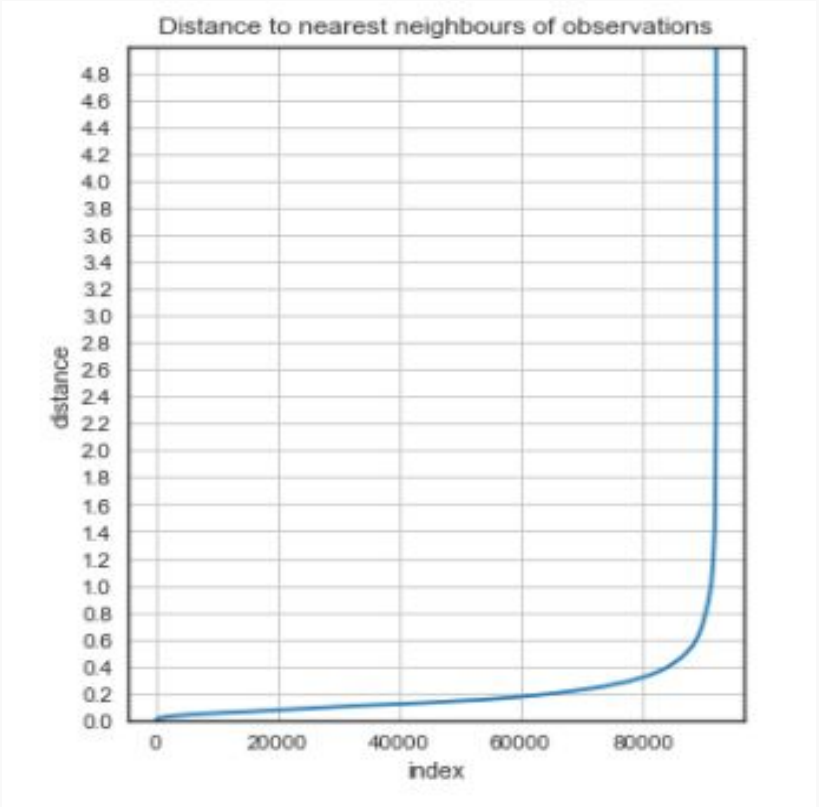
Sur la représentation t-SNE, on voit des groupes bien distincts par cluster malgré un peu de dispersion.
La séparation des clusters n'est pas forcément représentative sur une représentation t-sne.

3ème segmentation : DBSCAN

On conserve les mêmes variables.

Choix de l'hyperparamètre epsilon:

epsilon=0.5



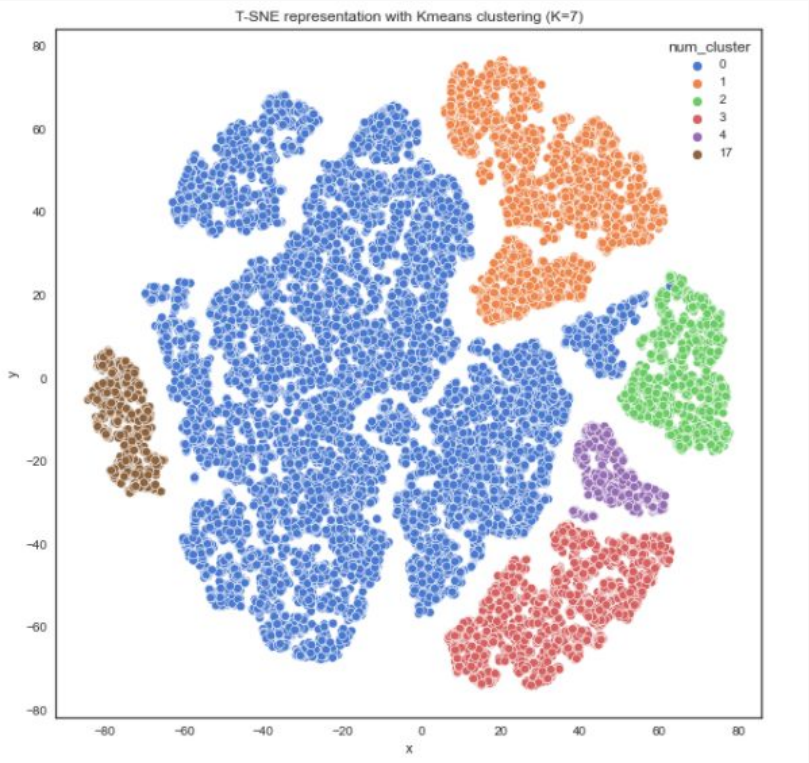
3ème segmentation : DBSCAN

```
dbscan=DBSCAN(eps=0.5,min_samples=4)
dbscan.fit(X_norm)
dg['num_cluster']=dbscan.labels_
u_4=dg['num_cluster'].value_counts()
```

207 clusters, 7000 observations en bruit (cluster -1)

On sélectionne les clusters ayant plus de 1500 observations -> 73000 observations récupérées

3ème segmentation : DBSCAN



On observe des clusters bien séparés mais beaucoup de pertes de données -> choix de la deuxième segmentation

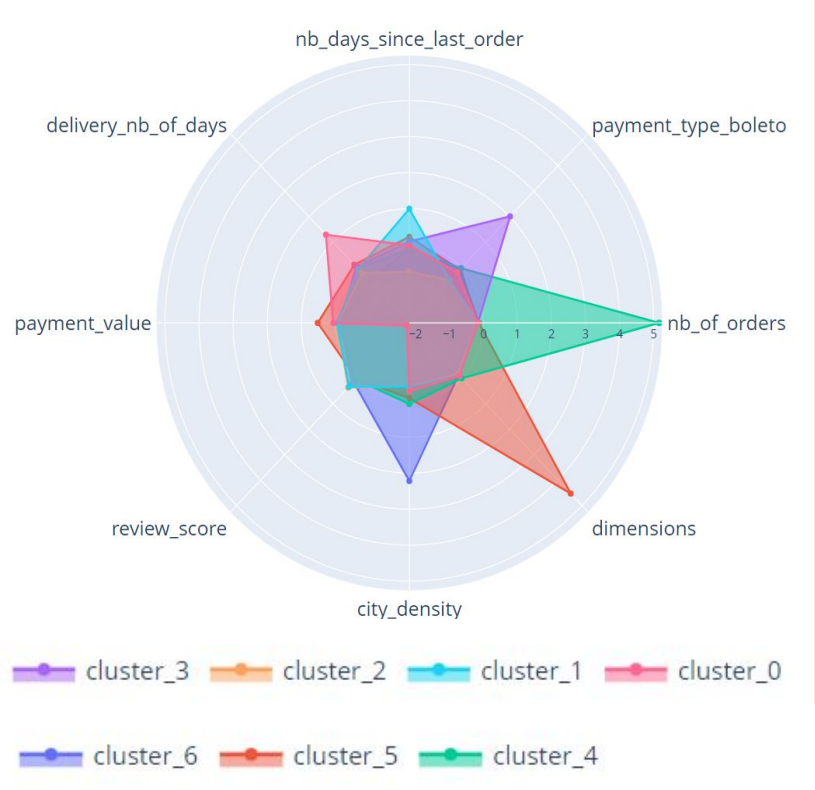
04 Conclusions

2e segmentation KMeans

K=7

	nb_of_orders	payment_type_boleto	nb_days_since_last_order	delivery_nb_of_days	payment_value	review_score	city_density	dimensions
0	1.000000	0.123000	233.312000	24.258000	172.986000	1.503000	1.775000	21865.765000
1	1.000000	0.000000	388.507000	11.546000	148.902000	4.575000	1.051000	21561.958000
2	1.000000	0.000000	121.623000	10.029000	146.241000	4.619000	0.984000	18564.841000
3	1.000000	1.000000	249.053000	12.305000	128.871000	4.420000	0.819000	19229.825000
4	2.115000	0.189000	218.888000	11.826000	143.206000	4.205000	3.793000	26449.650000
5	1.006000	0.166000	269.385000	13.090000	272.616000	4.171000	2.861000	247042.832000
6	1.000000	0.182000	219.526000	6.881000	128.062000	4.333000	16.171000	20058.177000

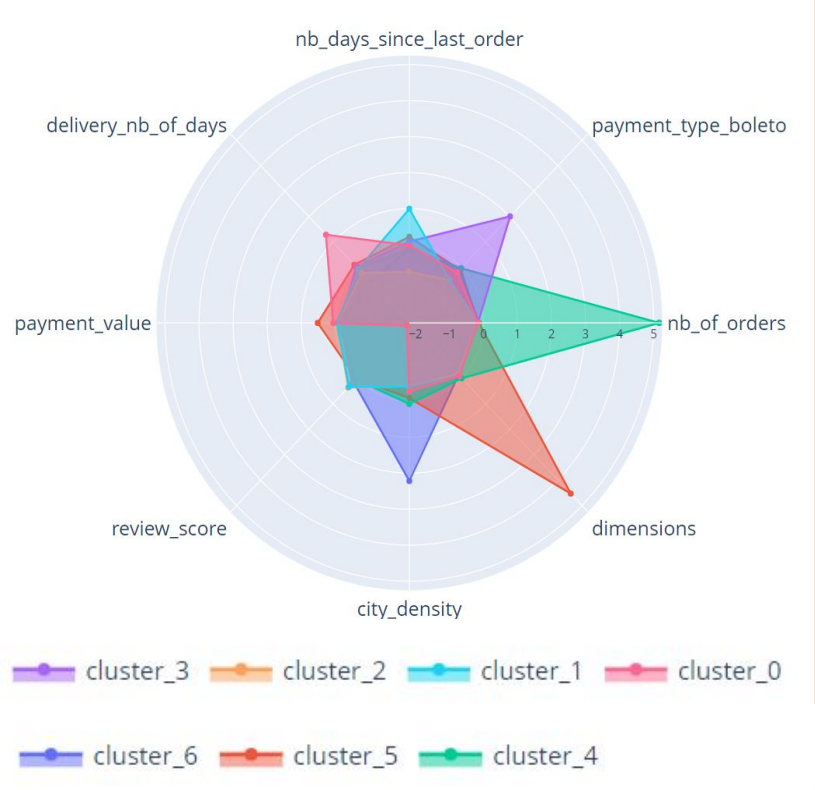
Comparaison des Clusters



Cluster 0

- ✓ 11 % de l'ensemble des clients
- ✓ Clients non satisfaits (1.5/5)
- ✓ Délais de livraison très longs
- ✓ Panier moyen : 170 réals
- ✓ Achat majoritaire : meubles&déco (23%)

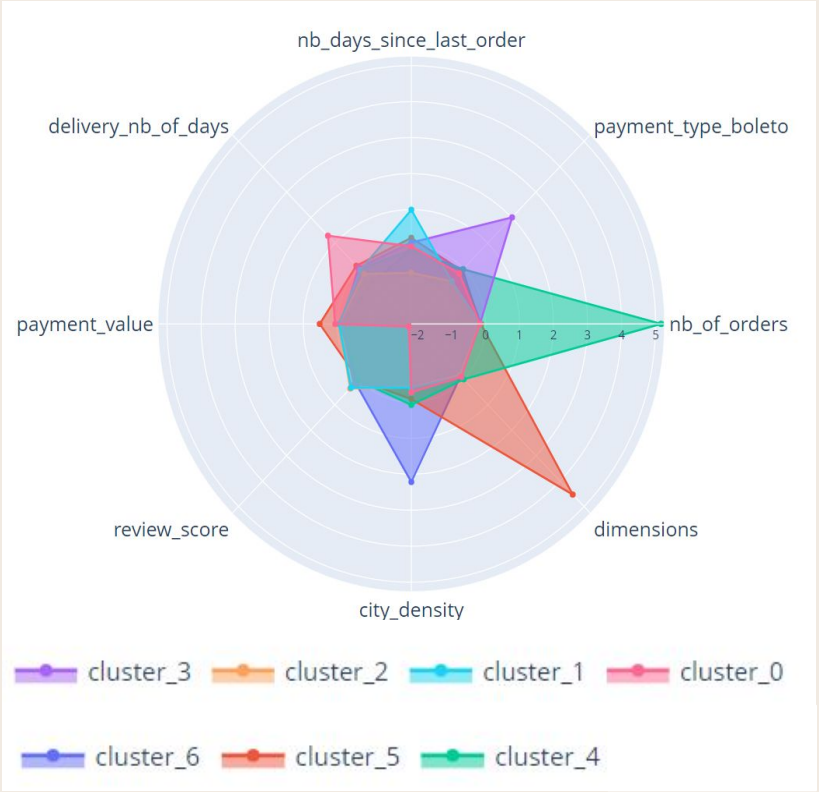
Comparaison des Clusters



Cluster 1

- ✓ 23 % de l'ensemble des clients
- ✓ Clients très satisfaits mais les moins récents
- ✓ Achat majoritaire : moblier&déco (20%), beauté&mode(20%)
- ✓ Panier moyen : 148 réals

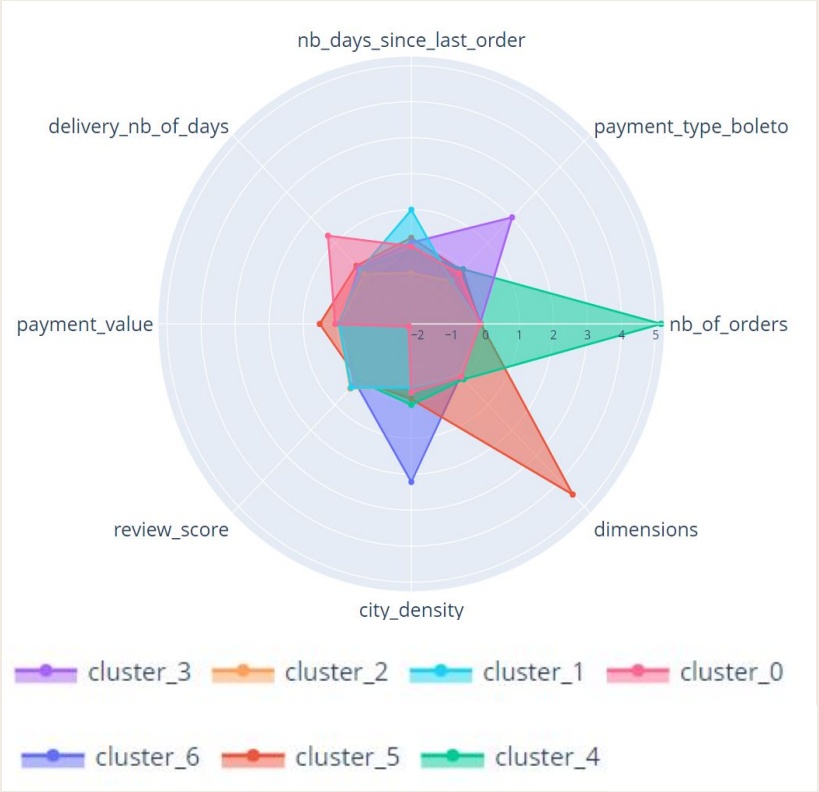
Comparaison des Clusters



Cluster 2

- ✓ 30 % de l'ensemble des clients
- ✓ Clients très satisfaits et les plus récents
- ✓ Achat majoritaire : mode&beauté (23%)
- ✓ Panier moyen : 146 réals

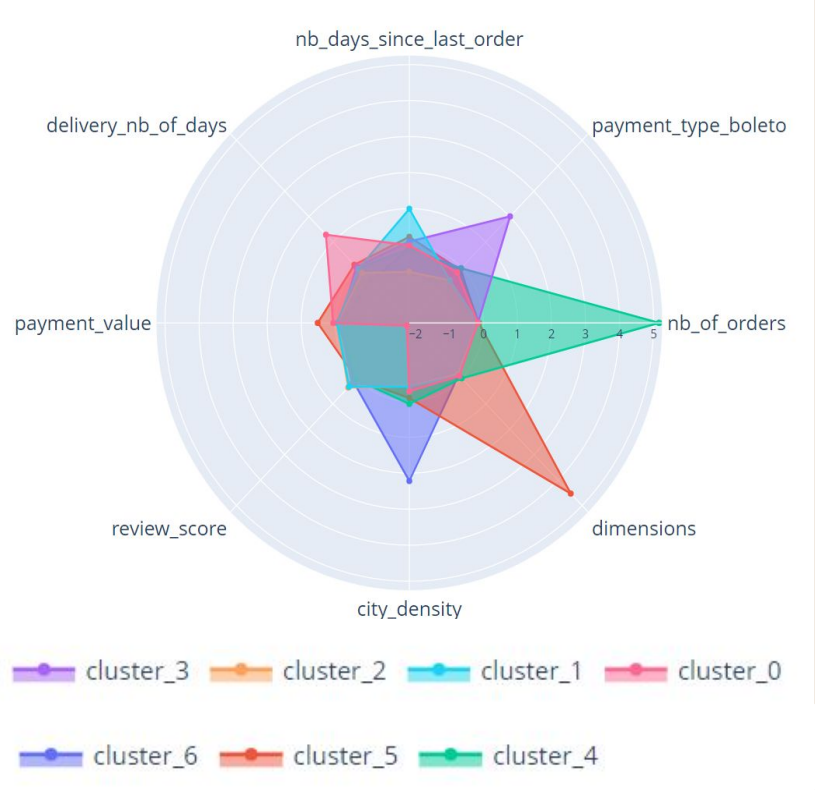
Comparaison des Clusters



Cluster 3

- ✓ 15 % de l'ensemble des clients
- ✓ Clients satisfaits
- ✓ Paiement par boleto
- ✓ Viennent de zones à faible densité de clients
- ✓ Panier moyen : 128 réals
- ✓ Achats majoritaire: électronique (18%)

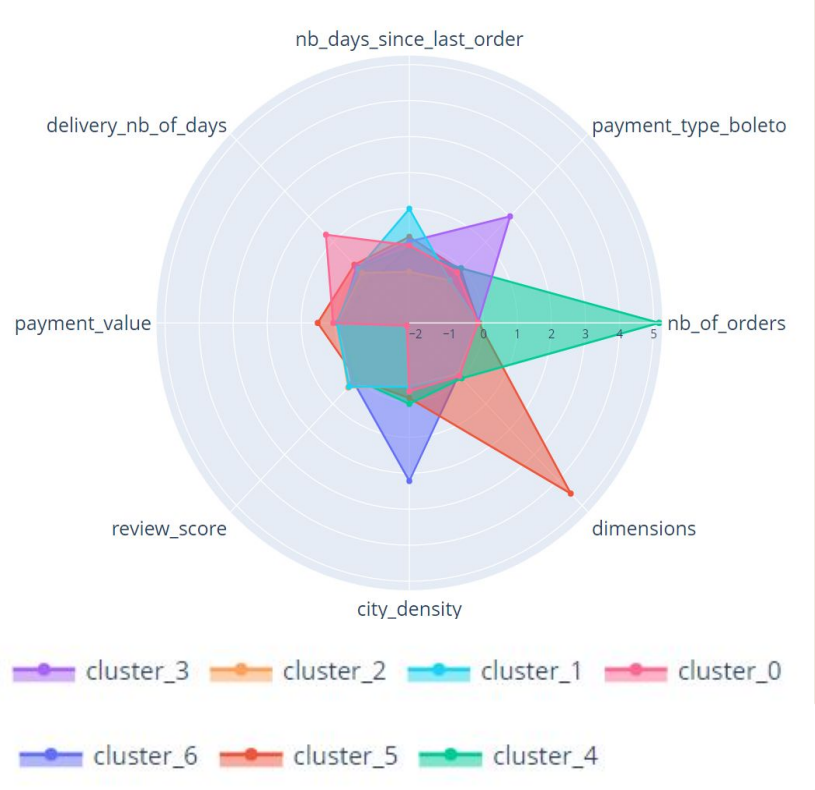
Comparaison des Clusters



Cluster 4

- ✓ 3 % de l'ensemble des clients
- ✓ Clients assez satisfaits
- ✓ Clients qui ont passé le plus de commandes
- ✓ Achats majoritaire: meubles&déco (27%)
- ✓ Panier moyen: 143 réals

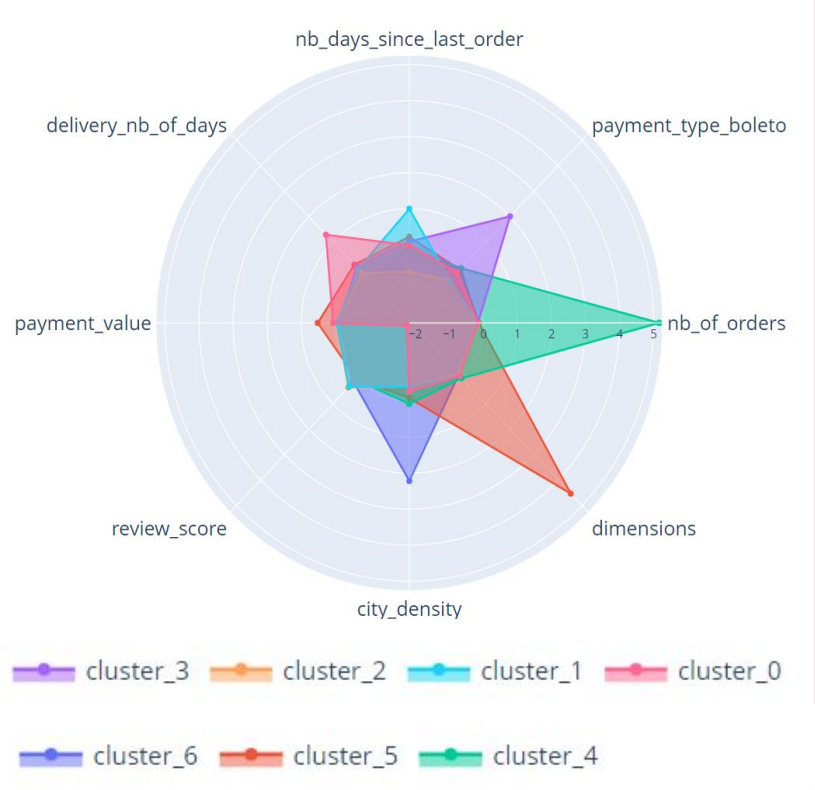
Comparaison des Clusters



Cluster 5

- ✓ 3 % de l'ensemble des clients
- ✓ Clients satisfaits
- ✓ Achat majoritaire : mobilier&déco (42%)
- ✓ Valeur de paiement la + élevée
- ✓ Dimensions des commandes plus élevée
- ✓ Panier moyen : 272 réals

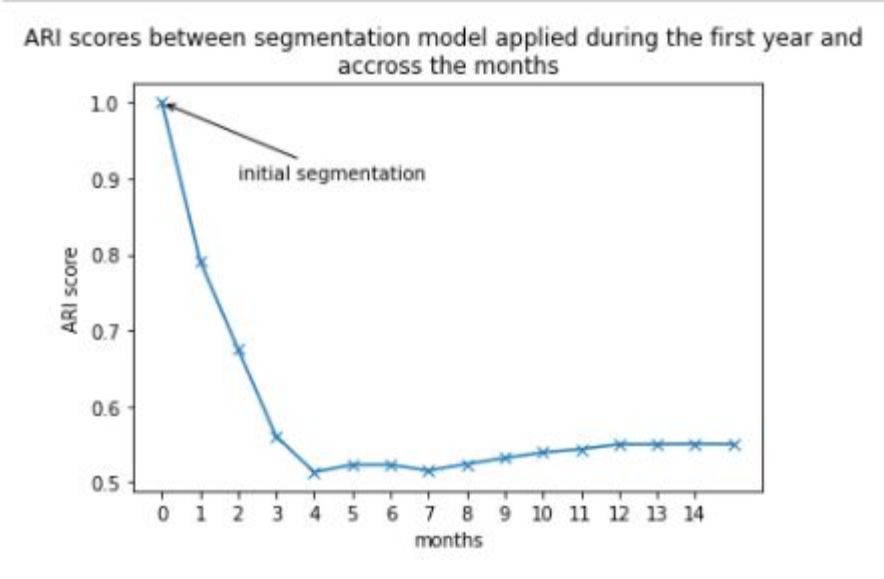
Comparaison des Clusters



Cluster 6

- ✓ 14 % de l'ensemble des clients
- ✓ Clients satisfaits habitant en ville (São Paulo)
- ✓ Courts délais de livraison
- ✓ Achat majoritaire : mode&beauté (20%) mobilier&déco (20%)
- ✓ Panier moyen : 128 réals

Contrat de maintenance



Proposition de renouvellement tous les deux mois

Annexe 1

```
def word_freq(data,col,nb=20):
    '''returns the most frequents words of more than three letters in the 'col' column
    '''
    word_count_dict = {}
    lp=list(data[col])
    word_count_dict=Counter()

    for i in range(data.shape[0]):
        text=str(lp[i]).lower()

        for w in text.split(" "):
            if len(w)>3:
                word_count_dict[w]+=1

    return word_count_dict.most_common(nb)
```

Annexe 2

```
dg_init=df.groupby('customer_unique_id').agg({#For each customer, we calculate
                                              # the number of orders
                                              'order_id': 'count',

                                               # the average nb of uses of each payment type
                                              'payment_type_boleto': 'mean',
                                              'payment_type_debit_card': 'mean',
                                              'payment_type_credit_card': 'mean',
                                              'payment_type_voucher': 'mean',

                                               # the average number of payments
                                              'nb_payments': 'mean',

                                               # the number of days since last order
                                              'nb_days_since_order': 'min',

                                               # the average delivery number of days
                                              'delivery_nb_of_days': 'mean',

                                               # the average payment value
                                              'payment_value': 'mean',

                                               # the average number of items
                                              'nb_items': 'mean',

                                               # the average review score
                                              'review_score': 'mean',

                                               # the customer's city density
                                               'city_density': lambda x: x.max(),

                                               # the average number of items for each category
                                               'category_other': 'mean',
                                               'category_kids': 'mean',
                                               'category_hobbies': 'mean',
                                               'category_furnitures_deco': 'mean',
                                               'category_food': 'mean',
                                               'category_fashion_beauty': 'mean',
                                               'category_electronic': 'mean',
                                               'category_diy': 'mean',
                                               'category_auto': 'mean',
                                               'category_services': 'mean',
                                               'category_accessories': 'mean',

                                               # the average dimension of items
                                              'dimensions': 'mean'
                                             })
```