OPENCLASSROOMS

Parcours Data Scientist

Projet 5 : Segmentez des clients d'un site e-commerce

Etudiant: Eric Wendling
Mentor: Julien Heiduk
Date: 07/04/2020

olist

Elaboration d'une stratégie marketing par segmentation de marché

Définition du projet

Contexte

- La société **olist** offre des services de commerce électronique en mode B2C via une plateforme de vente en ligne

Objectifs

- Optimisation des campagnes de communication marketing

Besoins

- Segmentation de marché (clients)

Moyens

- Base clients/commandes/produits/vendeurs
- Méthodes de segmentation (RFM, Clustering non supervisé)

Livrables

- Analyses exploratoires
- Evaluation RFM et modèles de Clustering

Plan

Analyse

- Identifier le domaine, les entités et leurs relations
- Analyse de l'intégrité du modèle et des données
  - Doublons, données manquantes…
- Construire un modèle de données adapté
  - Base de Clients avec leurs caractéristiques
  - Création de nouvelles variables

Extraction des caractéristiques des données
(Feature Engineering)

RFM

- Réalisation d'une segmentation RFM

Clustering

- Partionnement automatique non supervisée
  - K-Means / K-Prototypes / DBSCAN

Récupération des données

Données brutes

- Les données brutes sont disponibles sur le site: https://www.kaggle.com/olistbr/brazilian-ecommerce
- Support

  9 fichiers cvs

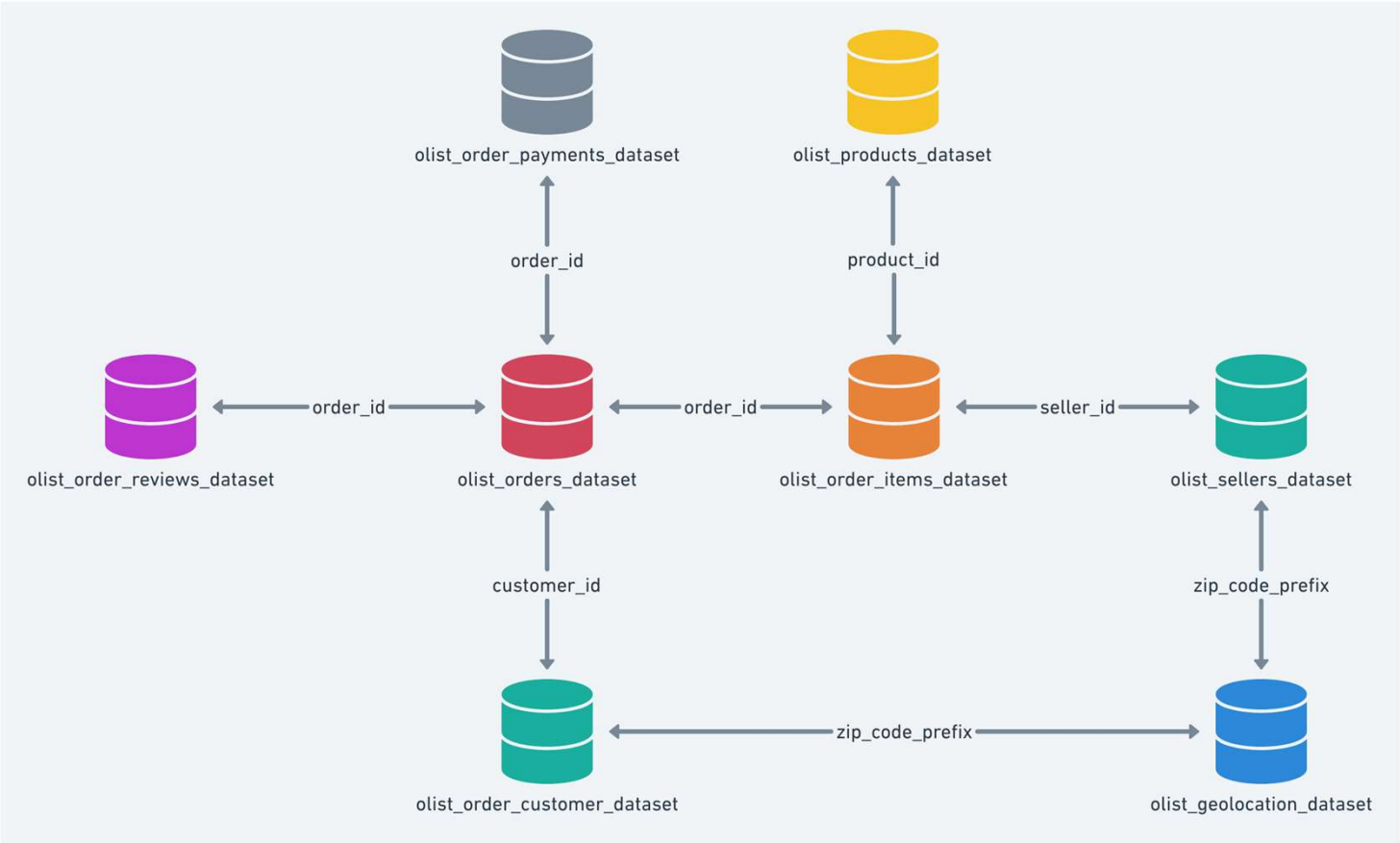
➢ Import des fichiers

  9 dataframes

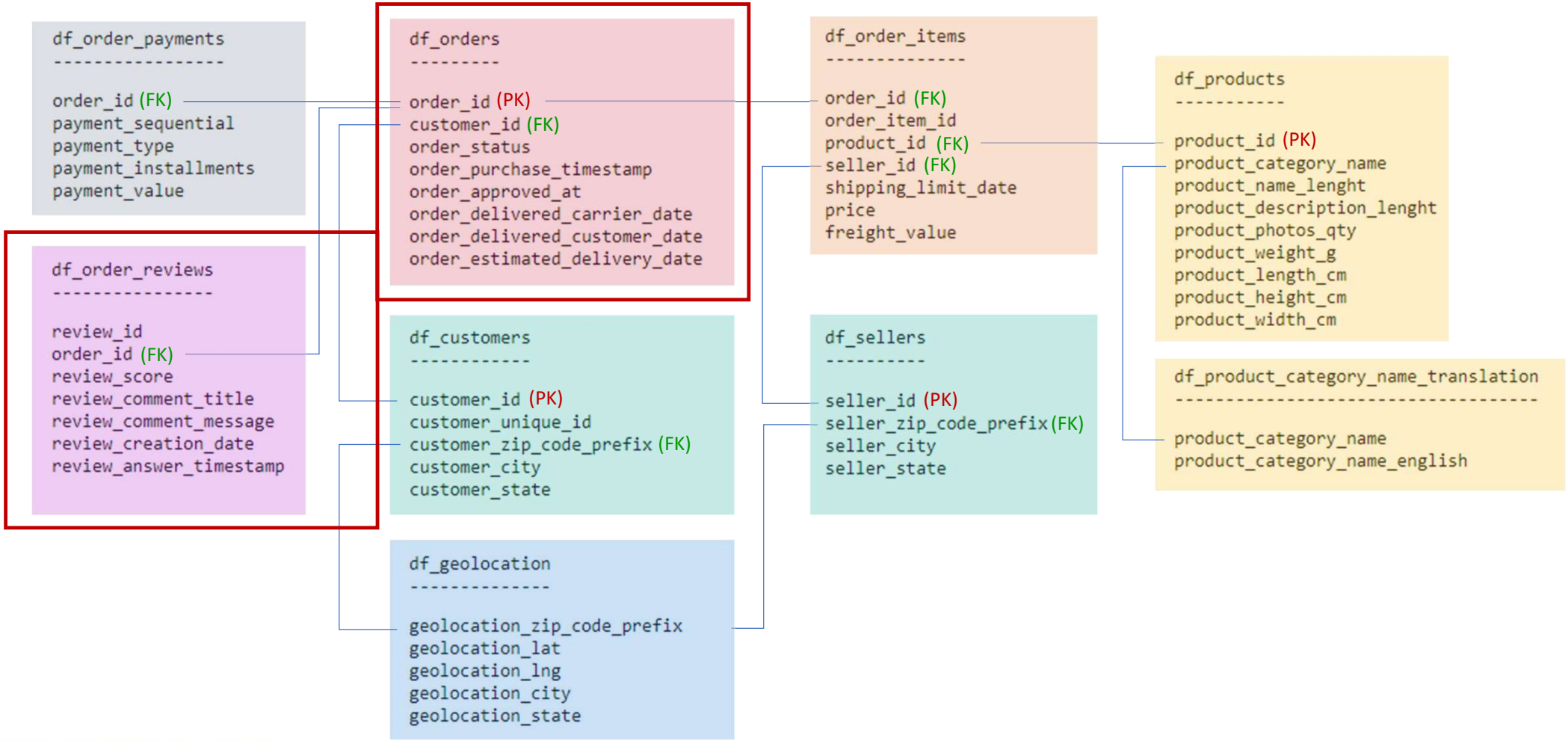
```
get_data = 0

if get_data == 0: # Import des données à partir des fichiers csv et stockage des dataframes

    df_customers = pd.read_csv('olist_customers_dataset.csv',encoding='utf-8',low_memory=False)
    df_geolocation = pd.read_csv('olist_geolocation_dataset.csv',encoding='utf-8',low_memory=False)
    df_order_items = pd.read_csv('olist_order_items_dataset.csv',encoding='utf-8',low_memory=False)
    df_order_payments = pd.read_csv('olist_order_payments_dataset.csv',encoding='utf-8',low_memory=False)
    df_order_reviews = pd.read_csv('olist_order_reviews_dataset.csv',encoding='utf-8',low_memory=False)
    df_orders = pd.read_csv('olist_orders_dataset.csv', encoding='utf-8',low_memory=False)
    df_products = pd.read_csv('olist_products_dataset.csv',encoding='utf-8',low_memory=False)
    df_sellers = pd.read_csv('olist_sellers_dataset.csv',encoding='utf-8',low_memory=False)
    df_product_category_name_translation = pd.read_csv('product_category_name_translation.csv',encoding='utf-8',low_memory=False

    %store df_customers
    %store df_geolocation
    %store df_order_items
    %store df_order_payments
    %store df_order_reviews
    %store df_orders
    %store df_products
    %store df_sellers
    %store df_product_category_name_translation
```

```
Durée de l'opération 'Récupération des données': 5.39 s
Durée totale de traitement: 00 h 00 m 05 s
```

Entités du modèle olist



Ensemble des datasets



Commandes / Evaluations

```
df_orders
---------

order_id (PK)
customer_id
order_status
order_purchase_timestamp
order_approved_at
order_delivered_carrier_date
order_delivered_customer_date
order_estimated_delivery_date
```

1 - N

1 - 1

```
df_order_reviews
----------------

review_id
order_id (FK)
review_score
review_comment_title
review_comment_message
review_creation_date
review_answer_timestamp
```

	Dataset	Shape	Variable	Duplicates	Unique	Null
0	orders	99441	order_id	0	99441	0
1	order_reviews	100000	review_id	827	99173	0
2	order_reviews	100000	order_id	559	99441	0

- A chaque enregistrement de ***orders*** correspond 1 ou plusieurs enregistrements de ***order_reviews***
- A chaque enregistrement de ***order_reviews*** correspond 1 et 1 seul enregistrement de ***orders***

Commandes / Evaluations

```
df_orders
---------

order_id (PK)
customer_id
order_status
order_purchase_timestamp
order_approved_at
order_delivered_carrier_date
order_delivered_customer_date
order_estimated_delivery_date
```

1 - N

```
df_order_reviews
----------------

review_id
order_id (FK)
review_score
review_comment_title
review_comment_message
review_creation_date
review_answer_timestamp
```

1 - 1

	Dataset	Shape	Variable	Duplicates	Unique	Null
0	orders	99441	order_id	0	99441	0
1	order_reviews	100000	review_id	827	99173	0
2	order_reviews	100000	order_id	559	99441	0
3	order_reviews_gr	99441	order_id	0	99441	0
4	orders	99441	order_id	0	99441	0

```
df_order_reviews_gr
-------------------

order_id
score_mean_per_ord
score_count_per_ord
```

1 - 1

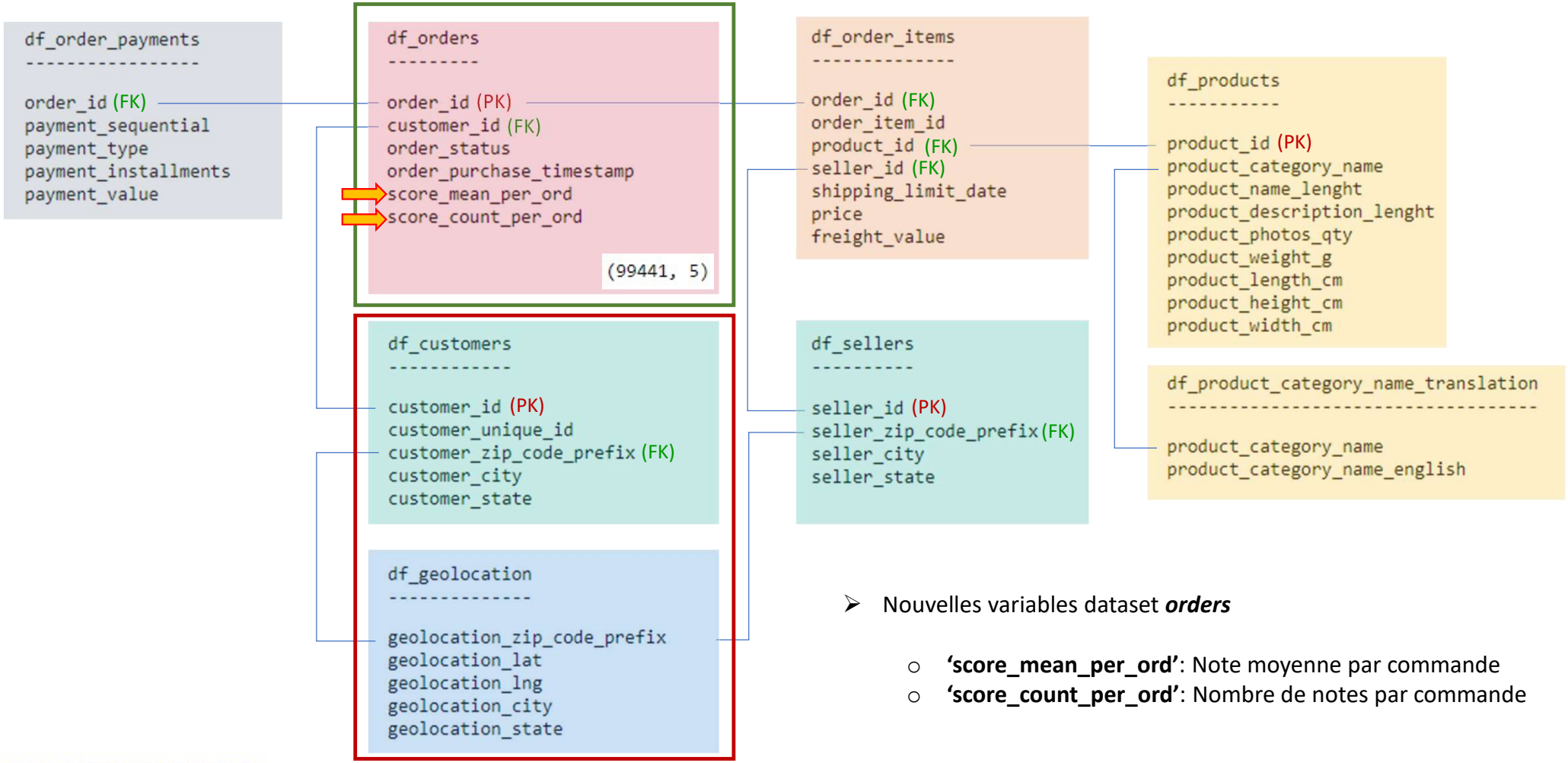
```
df_orders
---------

order_id
customer_id
order_status
order_purchase_timestamp
```

1 - 1

- A chaque enregistrement de ***orders*** correspond 1 ou plusieurs enregistrements de ***order_reviews***
- A chaque enregistrement de ***order_reviews*** correspond 1 et 1 seul enregistrement de ***orders***

1) On regroupe les notes par commande en faisant la moyenne des notes (**'review_score'**)
2) On fusionne les datasets ***orders*** et ***order_reviews_gr***

Ensemble des datasets



- Nouvelles variables dataset ***orders***
  - **'score_mean_per_ord'**: Note moyenne par commande
  - **'score_count_per_ord'**: Nombre de notes par commande

Clients / Geolocalisation

- La variable **'geolocation_zip_code_prefix'** contient des doublons

	Dataset	Shape	Variable	Duplicates	Unique	Null
4	geolocation	1000163	geolocation_zip_code_prefix	981148	19015	0

```
df_customers
------------

customer_id (PK)
customer_unique_id
customer_zip_code_prefix (FK)
customer_city
customer_state
```

0 - N

0 - N

```
df_geolocation
--------------

geolocation_zip_code_prefix  1000163 non-null
geolocation_lat              1000163 non-null
geolocation_lng              1000163 non-null
geolocation_city             1000163 non-null
geolocation_state            1000163 non-null
```

Clients / Geolocalisation

- La variable **'geolocation_zip_code_prefix'** contient des doublons

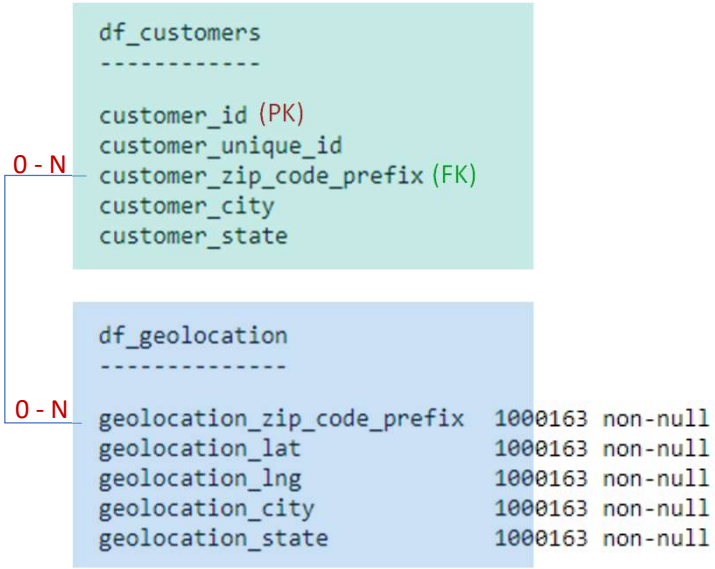
	Dataset	Shape	Variable	Duplicates	Unique	Null
4	geolocation	1000163	geolocation_zip_code_prefix	981148	19015	0
5	geolocation_gr_0	19015	geolocation_zip_code_prefix	0	19015	0

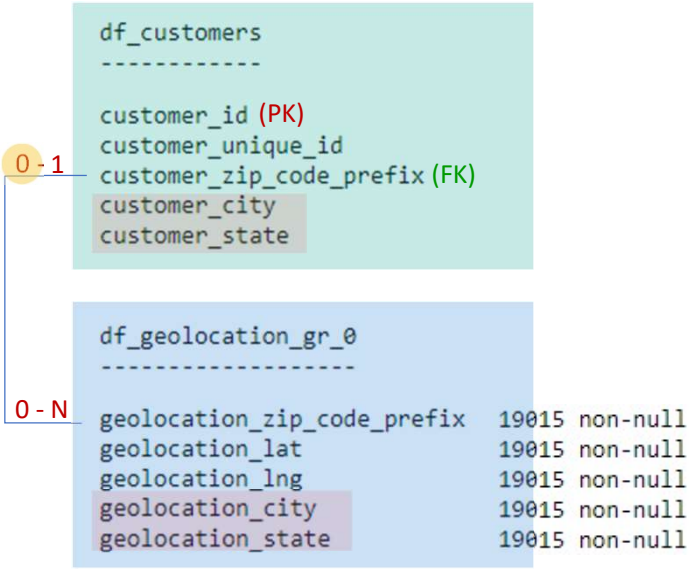
1) On regroupe les enregistrements sur la variable **'geolocation_zip_code_prefix'** et on réalise la moyenne des coordonnées géographiques

2) On fusionne les datasets ***customers*** et ***geolocation_gr_0***

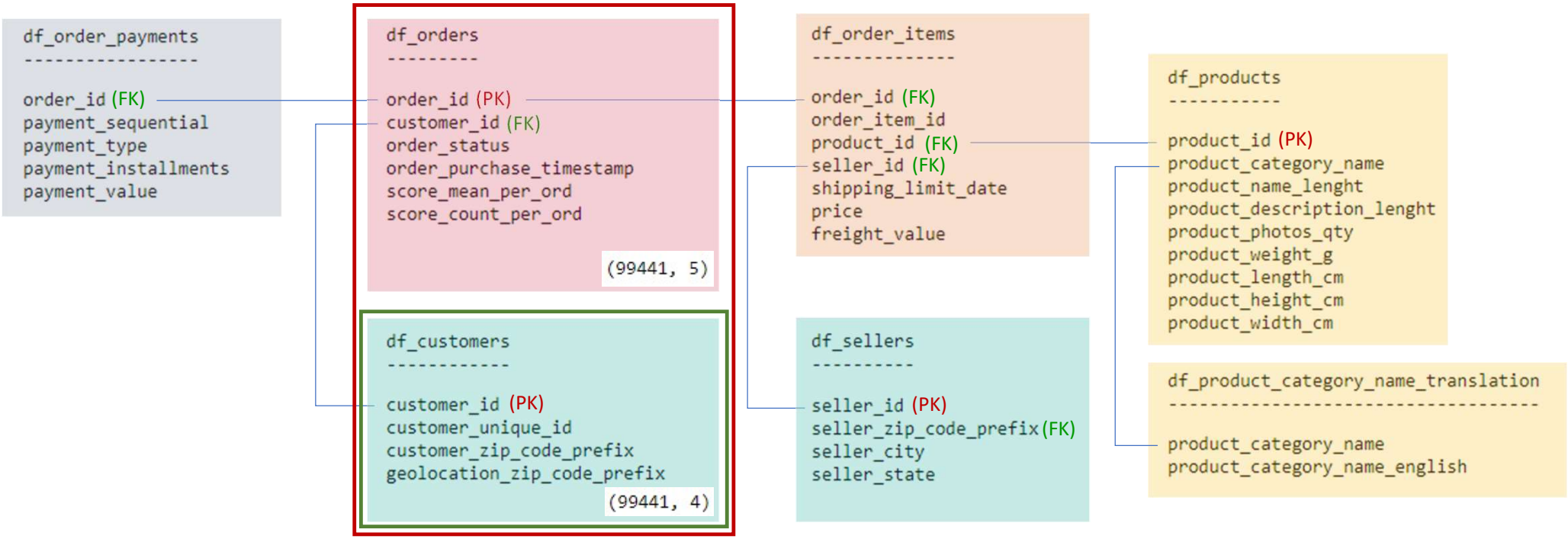
3) On compare les informations géographiques

- Les valeurs ne correspondent pas pour **337** références client
- ✓ Nous prenons comme référence les informations associées aux coordonnées géographiques





Ensemble des datasets



Clients / Commandes

```
df_orders
---------

order_id (PK)
customer_id (FK)
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord

(99441, 5)
```

1 - 1

1 - 1

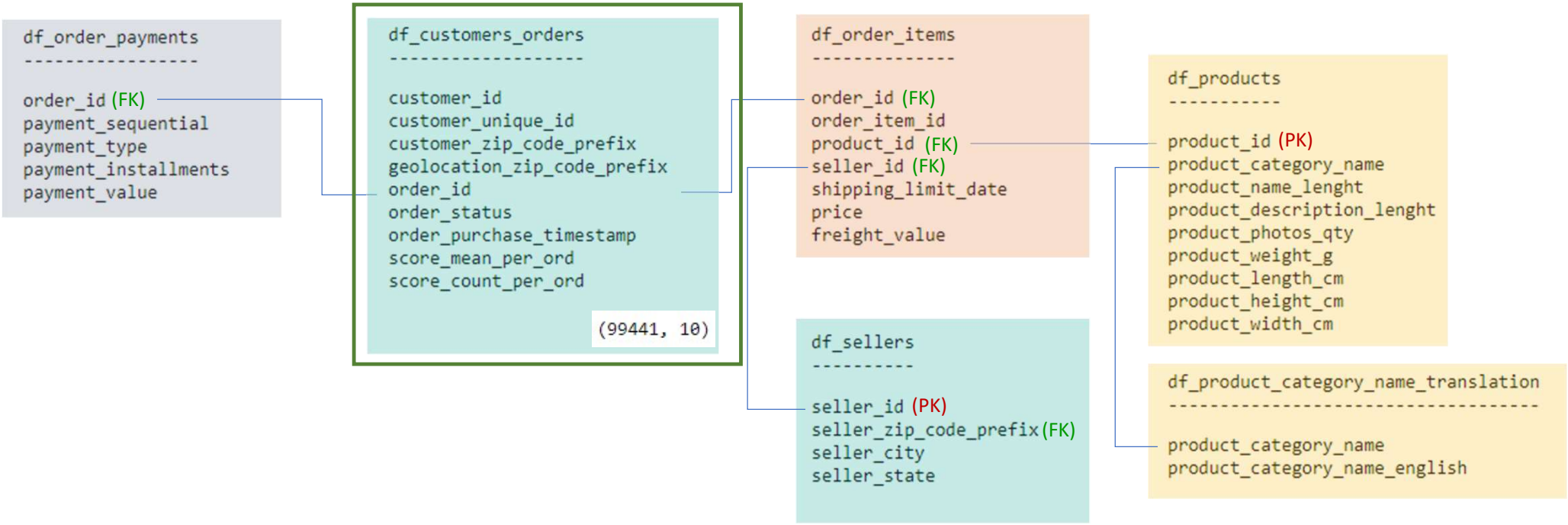
```
df_customers
------------

customer_id (PK)
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix
(99441, 4)
```

	Dataset	Shape	Variable	Duplicates	Unique	Null
7	customers	99441	customer_id	0	99441	0
8	customers	99441	customer_unique_id	3345	96096	0
9	orders	99441	order_id	0	99441	0
10	orders	99441	customer_id	0	99441	0

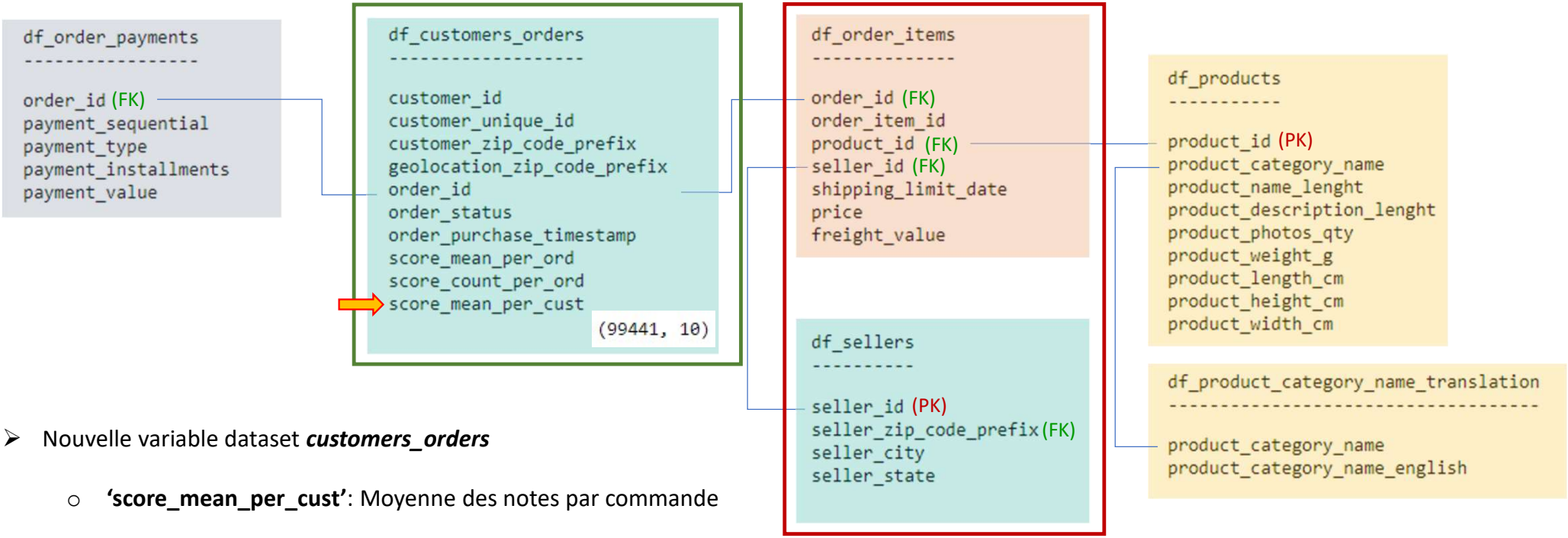
- La clé du dataset ***customers*** est la variable **'customer_id'** mais ne représente pas une valeur unique de client
- L'unicité du client est décrite par la variable **'customer_unique_id'**

1) On fusionne les datasets ***orders*** et ***customers***

Ensemble des datasets



Ensemble des datasets



- Nouvelle variable dataset ***customers_orders***
  - **'score_mean_per_cust'**: Moyenne des notes par commande

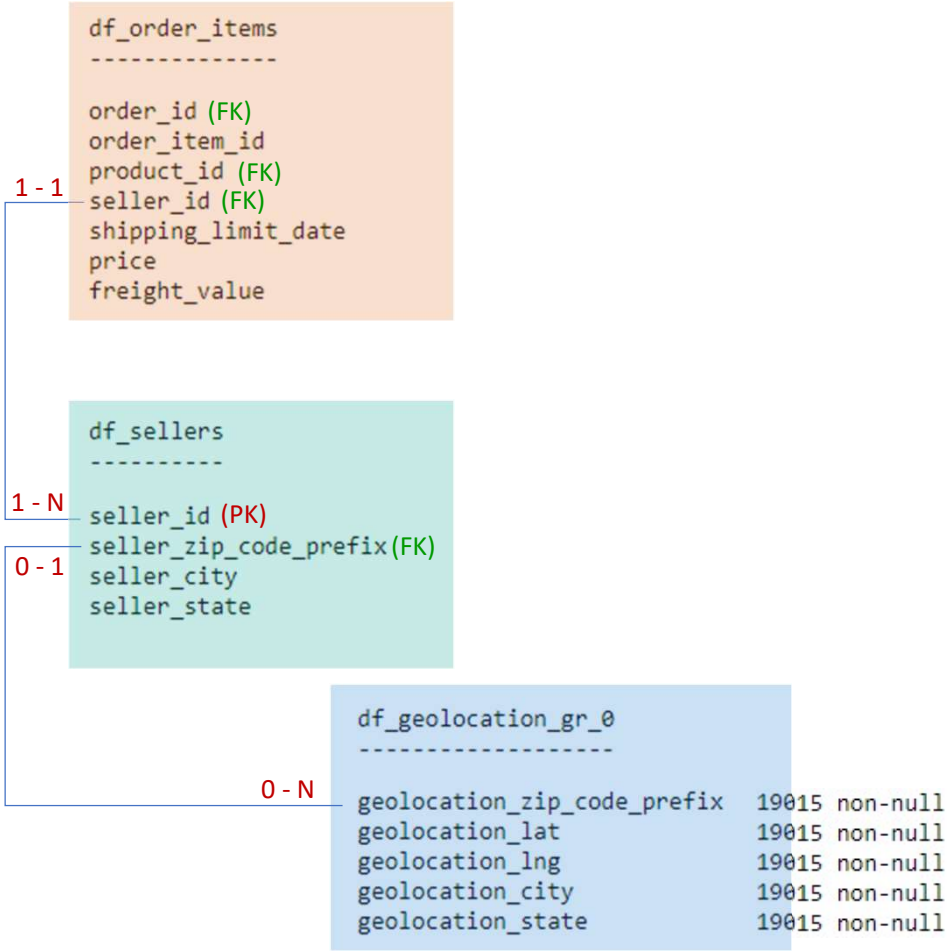
Lignes de commandes / Vendeurs

- A un enregistrement de ***order_items*** correspond 1 et 1 seul enregistrement de ***sellers***
- A un enregistrement de ***sellers*** correspondent 1 ou plusieurs enregistrements de ***order_items***

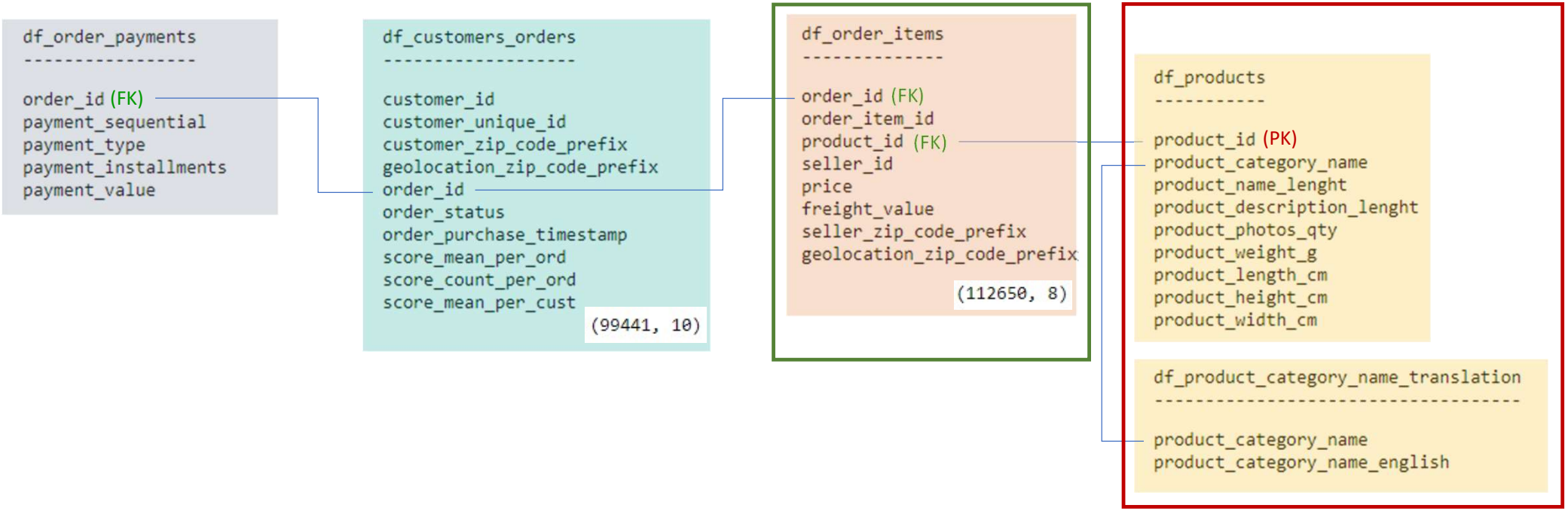
	Dataset	Shape	Variable	Duplicates	Unique	Null
12	order_items	112650	seller_id	109555	3095	0
13	sellers	3095	seller_id	0	3095	0

1) On fusionne les datasets ***sellers*** et ***geolocation_gr_0***

2) On fusionne les datasets ***order_items*** et ***sellers***



Ensemble des datasets



Produits / Catégories de produits

- A chaque enregistrement de ***products*** correspond 0 ou 1 enregistrement de ***product_category_name_translation***
- A chaque enregistrement de ***product_category_name_translation*** correspond 1 ou plusieurs enregistrements de ***products***

```
df_products
-----------

product_id (PK)
product_category_name
product_name_lenght
product_description_lenght
product_photos_qty
product_weight_g
product_length_cm
product_height_cm
product_width_cm
```

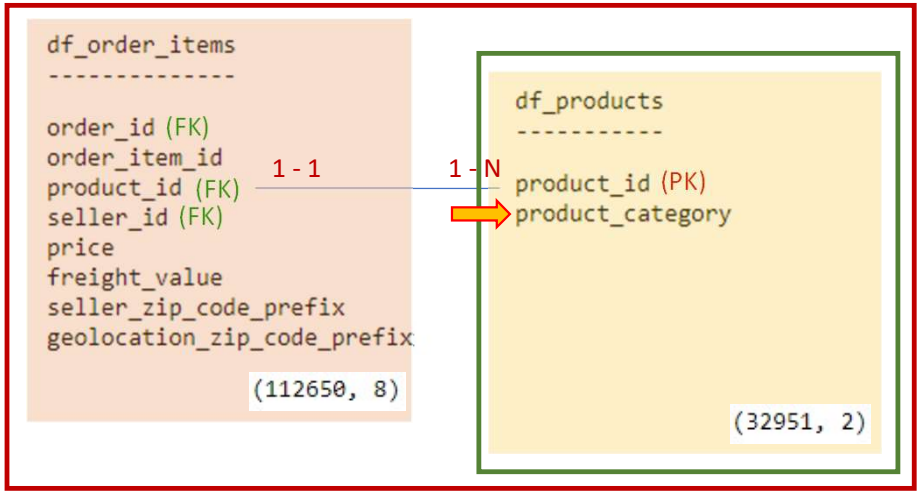
0 - 1

1 - N

```
df_product_category_name_translation
------------------------------------

product_category_name
product_category_name_english
```

1) On fusionne les datasets ***products*** et ***product_category_name_translation***

Lignes de commandes / Produits



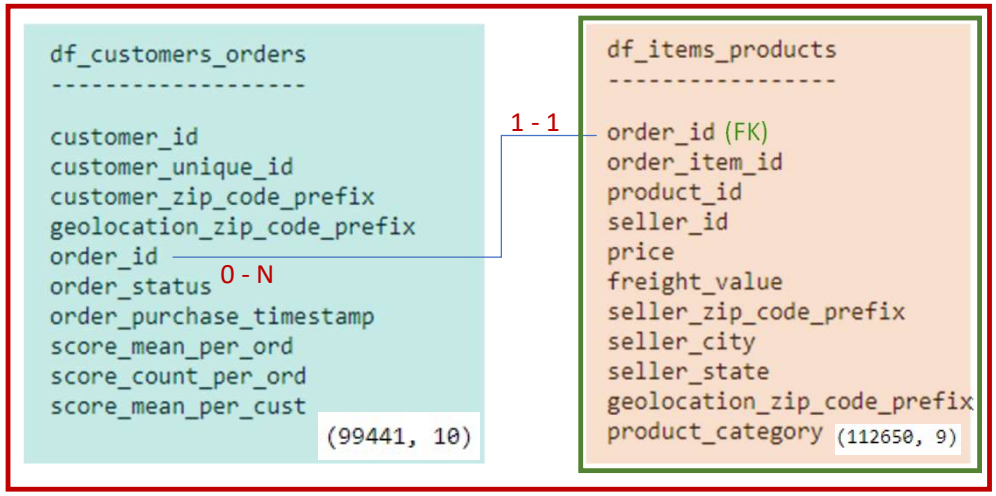
- A un enregistrement de ***order_items*** correspond 1 et 1 seul enregistrement de ***products***
- A un enregistrement de ***products*** correspondent 1 ou plusieurs enregistrements de ***order_items***

	Dataset	Shape	Variable	Duplicates	Unique	Null
19	order_items	112650	product_id	79699	32951	0
20	products	32951	product_id	0	32951	0

1) On fusionne les datasets ***order_items*** et ***products***

- Nouvelle variable dataset ***products***
  - **'product_category'**: Catégorie de produit (en anglais)

Clients / Commandes / Lignes de commandes / Produits



- A un enregistrement de ***customers_orders*** correspondent 0 ou plusieurs enregistrements de ***items_products***
- A un enregistrement de ***items_products*** correspond 1 et 1 seul enregistrement de ***customers_orders***

1) On fusionne les datasets ***customers_orders*** et ***items_products***

Ensemble des datasets

```
df_order_payments
-----------------

order_id (FK)
payment_sequential
payment_type
payment_installments
payment_value
```

```
df_customers_orders_products
----------------------------

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix_x
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
order_item_id
product_id
seller_id
price
freight_value
seller_zip_code_prefix
geolocation_zip_code_prefix_y
product_category
(112650, 18)
```

Clients / Commandes / Lignes de commandes / Produits

```
df_customers_orders_products
----------------------------

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix_x
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
order_item_id
product_id
seller_id
price
freight_value
seller_zip_code_prefix
geolocation_zip_code_prefix_y
product_category
price_total
price_per_ord
items_count
price_per_cust
prod_nb_avg
(112650, 18)
```

Nouvelles variables

- Prix total par produit
  - **'price_total'** = **'price'** + **'freight_value'**
- Montant par commande
  - **'price_per_order'**: Somme des prix totaux des produits par commande
- Nombre de produits par commande
  - **'items_counts'**
- Montant moyen par client 'unique'
  - **'price_per_cust'**: Moyenne des montants par commande
- Nombre de produits par client 'unique'
  - **'prod_nb_avg'**: Moyenne du nombre de produits par commande

Commandes / Paiements

```
df_order_payments
-----------------

order_id (FK)
payment_sequential
payment_type
payment_installments
payment_value
```

1 - 1

0 - N

```
df_customers_orders
-------------------

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
                    (99441, 10)
```

```
df_order_payments_gr
--------------------

order_id (FK)
payment_sum_per_ord
payment_count_per_ord
```

1 - 1

0 - 1

```
df_customers_orders
-------------------

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
                    (99441, 10)
```

	Dataset	Shape	Variable	Duplicates	Unique	Null
22	customers_orders	99441	order_id	0	99441	0
23	order_payments	103886	order_id	4446	99440	0
24	order_payments_gr	99440	order_id	0	99440	0

- A chaque enregistrement de ***customers_orders*** correspond 0 ou plusieurs enregistrements de ***order_payments***
- A chaque enregistrement de ***order_payments*** correspondent 1 et 1 seul enregistrement de ***customers_orders***

1) On regroupe les paiements par commande en faisant la somme des paiements
2) On fusionne les datasets ***customers_orders_products*** et ***order_payments_gr***

Clients / Commandes / Produits / Paiements

```
df_customers_orders_products_payments
-------------------------------------

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix_x
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
order_item_id
product_id
seller_id
seller_zip_code_prefix
geolocation_zip_code_prefix_y
product_category
price_total
price_per_ord
items_count
price_per_cust
prod_nb_avg
payment_sum_per_ord
payment_count_per_ord
(112650, 23)
```

- Nouvelles variables dataset ***customers_orders_products_payments***
  - **'payment_sum_per_ord'**: Montant payé par commande
  - **'payment_count_per_ord'**: Nombre de paiements par commande

Commandes / Paiements

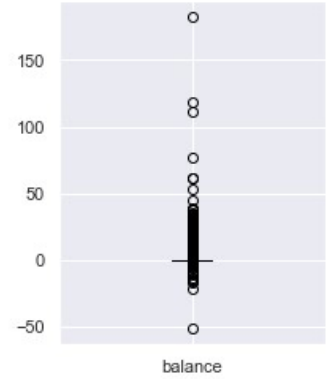
Balance des paiements

- Montants par Commande vs Paiement

order_id	payment_sum_per_ord	price_per_ord	balance
262118ce178bb3e4590a3adcf6d62e6b	126.12	177.74	-51.62
fd33085945f15975375cd8ec85440511	212.82	234.62	-21.80

order_id	payment_sum_per_ord	price_per_ord	balance
ce6d150fb29ada17d2082f4847107665	1586.47	1403.66	182.81
6e5fe7366a2e1bfbf3257dba0af1267f	406.92	287.91	119.01

	balance
count	98665.000000
mean	0.029092
std	1.129221
min	-51.620000
25%	-0.000000
50%	0.000000
75%	-0.000000
max	182.810000



Paiement supérieur au montant de la commande ?

Type de paiements

- Modes de paiement

payment_type	order_id
boleto	19784
credit_card	76795
debit_card	1529
not_defined	3
voucher	5775

Localisations

df = df_customers_orders_products_payments

```
df
--

customer_id
customer_unique_id
customer_zip_code_prefix
geolocation_zip_code_prefix_x
order_id
order_status
order_purchase_timestamp
score_mean_per_ord
score_count_per_ord
score_mean_per_cust
order_item_id
product_id
seller_id
seller_zip_code_prefix
geolocation_zip_code_prefix_y
product_category
price_total
price_per_cust
prod_nb_avg
payment_sum_per_ord
payment_count_per_ord

(112650, 23)
```

Localisation des clients uniques ?

Localisation de livraison

Localisation du vendeur

Localisations

Localisation des clients uniques

Localisations livraison vs vendeurs

customer_unique_id	same_loc	customer_u_zip_code_prefix	score_mean_per_cust	prod_nb_avg	price_per_cust
cust_1	0,33	12952	4,50	4	8 167

CLIENT

order_id	customer_zip_code_prefix	score_mean_per_ord	items_count	price_per_ord	payment_sum_per_ord
order_1	28013	5	5	7 600	5 500
order_2	12952	4,5	3	4 400	5 200
order_3	12952	4	4	12 500	6 550

COMMANDES

order_item_id	product_id	product_category	price_total	seller_id	seller_zip_code_prefix	same_loc
1	prod_1	audio	1 200	seller_1	11450	False
2	prod_1	audio	1 200	seller_1	11450	False
3	prod_1	audio	1 200	seller_1	11450	False
4	prod_2	toys	2 000	seller_3	14403	False
5	prod_2	toys	2 000	seller_3	14403	False
1	prod_1	audio	1 200	seller_2	12952	True
2	prod_1	audio	1 200	seller_2	12952	True
3	prod_2	toys	2 000	seller_2	12952	True
1	prod_2	toys	2 000	seller_2	12952	True
2	prod_3	food	3 500	seller_3	14403	False
3	prod_3	food	3 500	seller_3	14403	False
4	prod_3	food	3 500	seller_3	14403	False

LIGNES DE COMMANDES (PRODUITS)

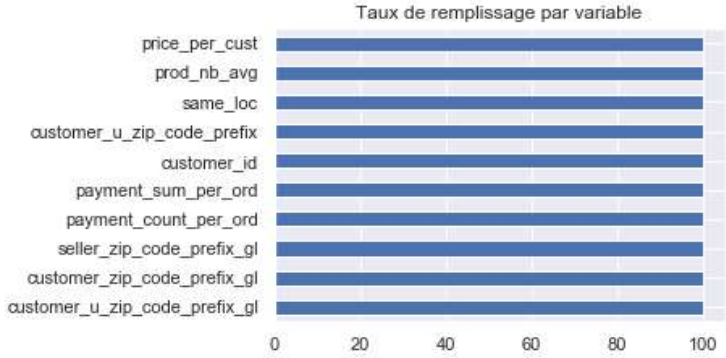
Montant total payé par commande
Moyenne des montants par commande
Somme des prix des produits d'une commande
Moyenne du nombre de produits par commande
Nombre de produits par commande
Moyenne des notes par commande
Note moyenne par commande
Principale localisation de livraison par client
Localisation de livraison par commande
Localisation du vendeur
Pourcentage de produits livrés à la même localisation que le vendeur

df *(Clients / Commandes / Produits / Paiements)*

```
df
--

customer_id                     112650 non-null object
customer_unique_id              112650 non-null object
customer_zip_code_prefix        112650 non-null int64
customer_zip_code_prefix_gl     112348 non-null float64
order_id                        112650 non-null object
order_status                    112650 non-null object
score_mean_per_ord              112650 non-null float64
score_count_per_ord             112650 non-null int64
score_mean_per_cust             112650 non-null float64
order_item_id                   112650 non-null int64
product_id                      112650 non-null object
seller_id                       112650 non-null object
seller_zip_code_prefix          112650 non-null int64
seller_zip_code_prefix_gl       112397 non-null float64
product_category                112650 non-null object
price_total                     112650 non-null float64
price_per_ord                   112650 non-null float64
items_count                     112650 non-null int64
price_per_cust                  112650 non-null float64
prod_nb_avg                     112650 non-null float64
payment_sum_per_ord             112647 non-null float64
payment_count_per_ord           112647 non-null float64
same_loc                        112650 non-null bool
customer_u_zip_code_prefix      112650 non-null int64
customer_u_zip_code_prefix_gl   112347 non-null float64
purchase_date                   112650 non-null datetime64[ns]
```

Changement format date

```
Affichage des 10 premières variables triées par taux croissant:

                                Taux de remplissage (%)
customer_u_zip_code_prefix_gl                     99.731
customer_zip_code_prefix_gl                       99.732
seller_zip_code_prefix_gl                         99.775
payment_count_per_ord                             99.997
payment_sum_per_ord                               99.997
customer_id                                      100.000
customer_u_zip_code_prefix                       100.000
same_loc                                         100.000
prod_nb_avg                                      100.000
price_per_cust                                   100.000
```



df1 *(Niveau Clients)*

```
df1 = df.groupby('customer_unique_id').agg({'purchase_date': ['min','max'],
                                            'order_id': pd.Series.nunique,
                                            'product_id': 'count',
                                            'price_total': ['mean','sum'],
                                            'price_per_cust': 'first',
                                            'prod_nb_avg': 'first',
                                            'score_mean_per_ord': 'mean',
                                            'score_mean_per_cust': 'first',
                                            'same_loc': 'sum',
                                            'customer_u_zip_code_prefix': 'first'})
```

	purchase_date		order_id	product_id	price_total		price_per_cust	prod_nb_avg	score_mean_per_ord
	min	max	nunique	count	mean	sum	first	first	mean
customer_unique_id									
8d50f5eadf50201ccdcedfb9e2ac8455	2017-05-15	2018-08-20	16	16	56.377500	902.04	56.377500	1.000000	5.000000
3e43e6105506432c953e165fb2acf44c	2017-09-18	2018-02-27	9	14	83.762143	1172.67	130.296667	1.555556	2.642857
1b6c7548a2a1f9037c1fd3ddfed95f33	2017-11-13	2018-02-14	7	9	106.556667	959.01	137.001429	1.285714	5.000000

```
df1.shape
```

```
(95420, 12)
```

Variables catégorielles

Niveau Lignes de commandes

- Variable **'Product_Category'**
- OneHotEncoder
  - 74 valeurs différentes

	customer_unique_id	product_category	x0_bed_bath_table	x0_musical_instruments	x0_toys	x0_audio
0	0000366f3b9a7992bf8c76cfdf3221e2	bed_bath_table	1.0	0.0	0.0	0.0
1	000fbf0473c10fc1ab6f8d2d286ce20c	musical_instruments	0.0	1.0	0.0	0.0
2	000fbf0473c10fc1ab6f8d2d286ce20c	musical_instruments	0.0	1.0	0.0	0.0
3	000fbf0473c10fc1ab6f8d2d286ce20c	toys	0.0	0.0	1.0	0.0
4	000fbf0473c10fc1ab6f8d2d286ce20c	toys	0.0	0.0	1.0	0.0

```
df.shape
```

```
(112650, 100)
```

Niveau Clients

- On regroupe les données par client unique et on agrège sur la valeur max de la variable catégorie de produit
- Ainsi, on assigne à chaque client les catégories de produits qu'il a commandé au moins une fois

customer_unique_id	order_id	order_item_id	product_id	product_category
cust_1	order_1	1	prod_1	audio
		2	prod_1	audio
		3	prod_1	audio
		4	prod_2	toys
		5	prod_2	toys
	order_2	1	prod_1	audio
		2	prod_1	audio
		3	prod_2	toys
	order_3	1	prod_2	toys
		2	prod_3	food
		3	prod_3	food
		4	prod_3	food
CLIENT	COMMANDES	LIGNES DE COMMANDES (PRODUITS)		

Nombre de produits différents

	audio	toys	food	health	telephony	
Max	1	1	1	0	0	3
	1	0	0	0	0	
	1	0	0	0	0	
	1	0	0	0	0	
	0	1	0	0	0	
	0	1	0	0	0	
	1	0	0	0	0	
	1	0	0	0	0	
	0	1	0	0	0	
	0	1	0	0	0	
	0	0	1	0	0	
	0	0	1	0	0	
	0	0	1	0	0	

customer_unique_id	x0_bed_bath_table	x0_musical_instruments	x0_toys	x0_audio
0000366f3b9a7992bf8c76cfdf3221e2	1.0	0.0	0.0	0.0
000fbf0473c10fc1ab6f8d2d286ce20c	0.0	1.0	1.0	0.0

```
df_cat.shape
```

```
(95420, 74)
```

- Nouvelle variable
  - **'nb_prod_diff'**: Nombre de produits différents par client

Variables RFM

df_rfm = df1[list_rfm]

```
(purchase_date, min)   2016-09-04
(purchase_date, max)   2018-09-03
```

(customer_unique_id,)	(purchase_date, min)	(purchase_date, max)	(order_id, nunique)	(product_id, count)	(price_total, mean)	(price_total, sum)	(price_per_cust, first)	(prod_nb_avg, first)	(score_mean_per_cust, first)	(same_loc, sum)	nb_prod_diff
8d50f5eadf50201ccdcedfb9e2ac8455	2017-05-15	2018-08-20	16	16	56.377500	902.04	56.377500	1.000000	4.764706	0.0	4
3e43e6105506432c953e165fb2acf44c	2017-09-18	2018-02-27	9	14	83.762143	1172.67	130.296667	1.555556	2.777778	0.0	5
1b6c7548a2a1f9037c1fd3ddfed95f33	2017-11-13	2018-02-14	7	9	106.556667	959.01	137.001429	1.285714	5.000000	0.0	5
6469f99c1f9dfae7733b25662e7f1782	2017-09-19	2018-06-28	7	9	84.314444	758.83	108.404286	1.285714	5.000000	0.0	1
ca77025e7201e3b30c44b472ff346268	2017-10-09	2018-06-01	7	12	93.560000	1122.72	160.388571	1.714286	5.000000	0.0	4

customer_unique_id	start	end	price_total	prod_nb_avg	duration	older	recency	frequency	frequency_2	monetary	same_loc	score
8d50f5eadf50201ccdcedfb9e2ac8455	2017-05-15	2018-08-20	902.04	1.000000	463.0	476.0	14.0	16	16	56.377500	0.0	4.764706
3e43e6105506432c953e165fb2acf44c	2017-09-18	2018-02-27	1172.67	1.555556	163.0	350.0	188.0	9	14	130.296667	0.0	2.777778
1b6c7548a2a1f9037c1fd3ddfed95f33	2017-11-13	2018-02-14	959.01	1.285714	94.0	294.0	201.0	7	9	137.001429	0.0	5.000000
6469f99c1f9dfae7733b25662e7f1782	2017-09-19	2018-06-28	758.83	1.285714	283.0	349.0	67.0	7	9	108.404286	0.0	5.000000
ca77025e7201e3b30c44b472ff346268	2017-10-09	2018-06-01	1122.72	1.714286	236.0	329.0	94.0	7	12	160.388571	0.0	5.000000

- **'duration'**: Nombre de jours entre la première et la dernière date d'achat par client
- **'older'**: Nombre de jours depuis la date d'achat la plus ancienne par client
- **'recency'**: Nombre de jours depuis la date d'achat la plus récente par client
- **'frequency'**: Fréquence d'achats par client
- **'monetary'**: Montants par client
- **'same_loc'**: Pourcentage de produits livrés à la même localisation que le vendeur

Outliers

Identifiés avec ACP

- Client 1
  - ID = 8d50f5eadf50201ccdcedfb9e2ac8455
  - Client très récent
  - Nombreuses commandes (16)
  - Nombreux produits (16)
  - 1 produit par commande
  - 4 produits différents
  - Panier moyen faible
- Client 2
  - ID = 0a0a92112bd4c708ca5fde585afaa872
  - Client moyennement récent
  - Peu de commande (1)
  - Nombreux produits (8)
  - 8 produits par commande
  - 1 seule catégorie de produits
  - Panier moyen très élevé

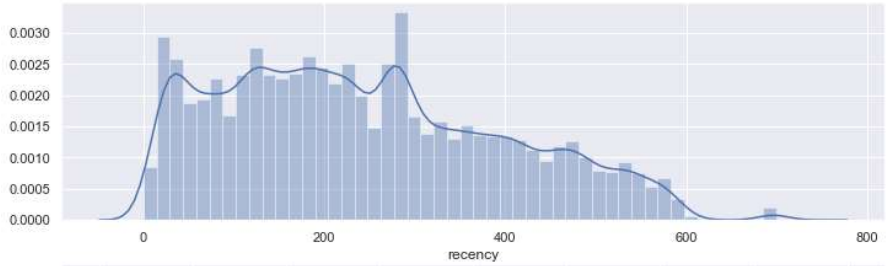
Client 1

customer_unique_id	8d50f5eadf50201ccdcedfb9e2ac8455
start	2017-05-15 00:00:00
end	2018-08-20 00:00:00
older	476
recency	14
duration	463
frequency (number of orders)	16
frequency_2 (number of products)	16
product number average	1
number of different products	4
monetary	56.3775
average price	56.3775
total price	902.04
seller loc = delivery loc	0
mean score	4.76471
R	5
F	5
M	2
RFM	552

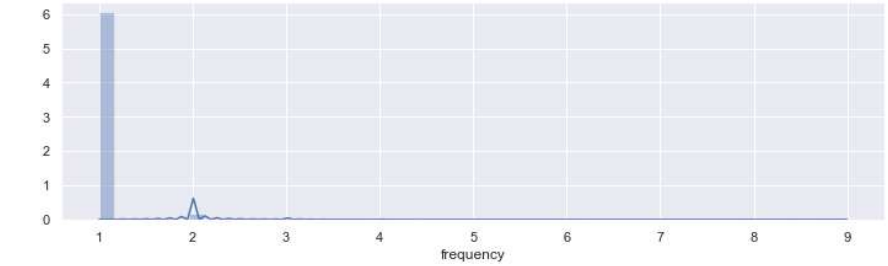
Client 2

customer_unique_id	0a0a92112bd4c708ca5fde585afaa872
start	2017-09-29 00:00:00
end	2017-09-29 00:00:00
older	339
recency	339
duration	1
frequency (number of orders)	1
frequency_2 (number of products)	8
product number average	8
number of different products	1
monetary	13664.1
average price	1708.01
total price	13664.1
seller loc = delivery loc	0
mean score	1
R	2
F	1
M	5
RFM	215

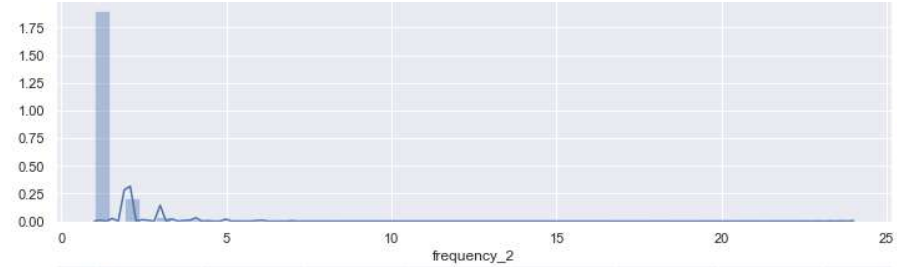
Distributions RFM



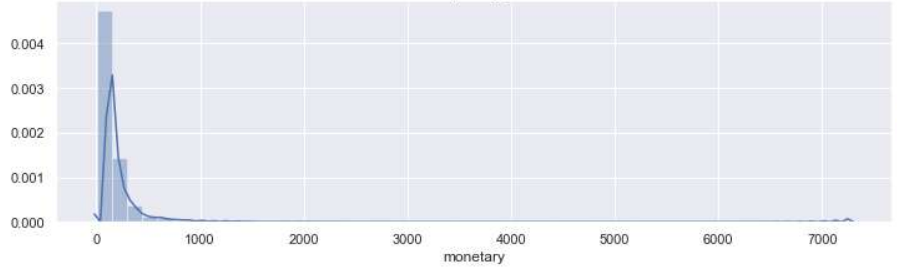
Coefficient d'asymétrie 'recency': 0.449



Coefficient d'asymétrie 'frequency': 8.368

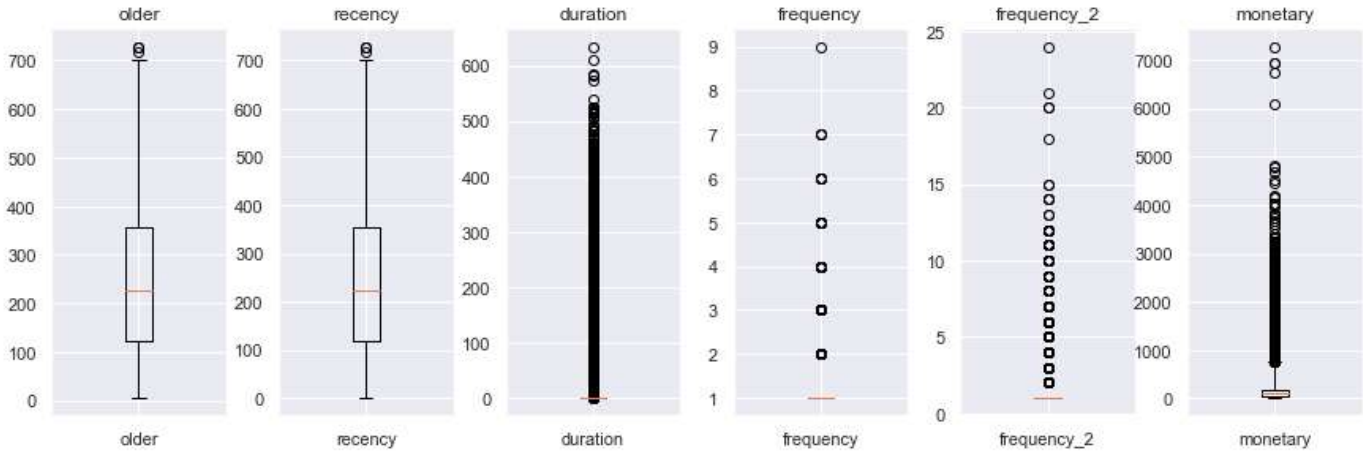


Coefficient d'asymétrie 'frequency_2': 7.334



Coefficient d'asymétrie 'monetary': 7.322

Statistiques RFM



	customer_unique_id
frequency	
1	92506
2	2673
3	192
4	29
5	9
6	5
7	3
9	1

Nombre de clients par nombre de commandes réalisées

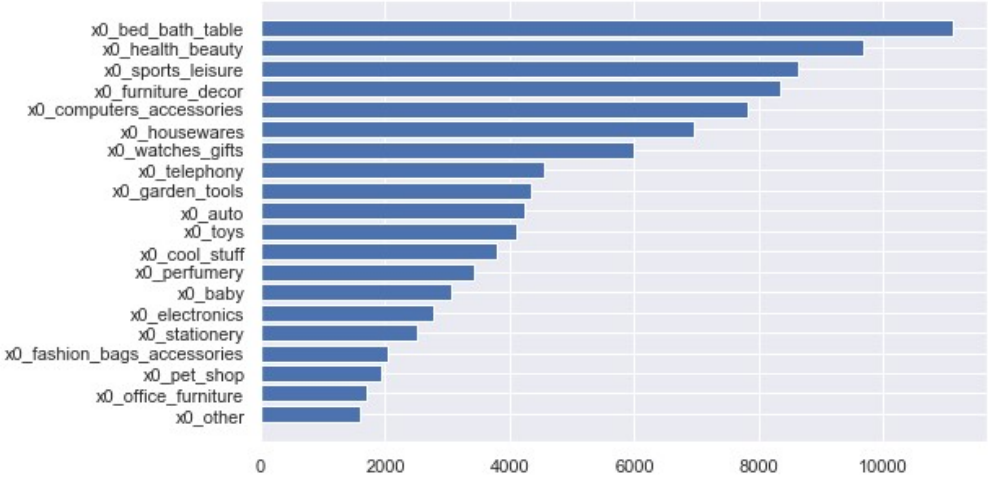
	older	recency	duration	frequency	frequency_2	monetary
count	95418.000000	95418.000000	95418.000000	95418.000000	95418.000000	95418.000000
mean	246.181465	243.502861	3.678604	1.033862	1.180343	160.904352
std	153.655917	153.160341	25.219539	0.205605	0.619208	216.831271
min	5.000000	0.000000	1.000000	1.000000	1.000000	9.590000
25%	122.000000	119.000000	1.000000	1.000000	1.000000	62.410000
50%	227.000000	224.000000	1.000000	1.000000	1.000000	105.770000
75%	356.000000	353.000000	1.000000	1.000000	1.000000	177.118750
max	729.000000	729.000000	634.000000	9.000000	24.000000	7274.880000

- **92506** clients ont réalisé une seule commande (fréquence = 1) ce qui représente **97.0%** de l'ensemble des clients.
- **83551** clients ont acheté un seul produit (fréquence_2 = 1) ce qui représente **88.0%** de l'ensemble des clients.

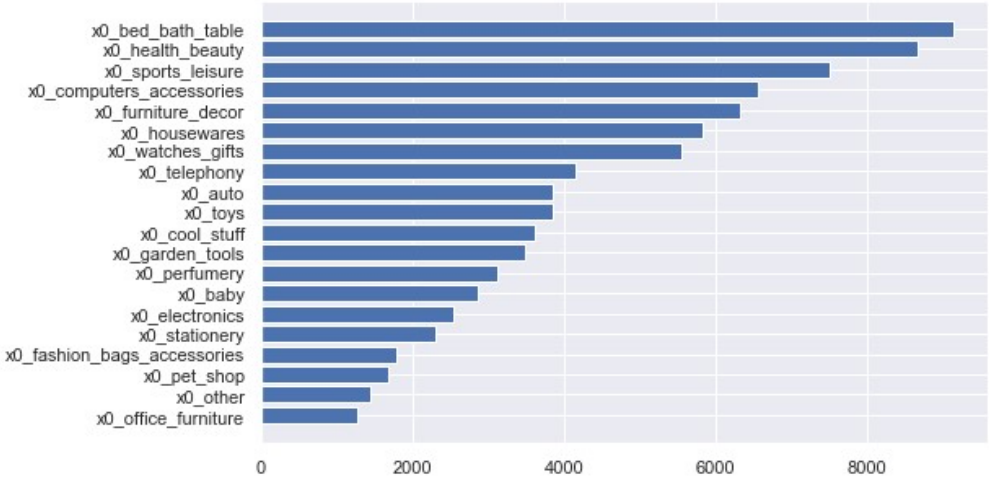
Statistiques RFM

Produits les plus vendus

- Niveau Lignes de commandes



- Niveau Clients



Segmentation RFM

Répartition des données par variable RFM

- 5 classes

```
nb_bins = 5

cl_r = pd.qcut(df_rfm['recency'], 5, labels = range(nb_bins, 0, -1)) # range de 5 à 1
cl_f = pd.cut(df_rfm['frequency'], bins = nb_bins, labels = range(1, nb_bins+1)) # range de 1 à 5
cl_m = pd.qcut(df_rfm['monetary'], 5, labels = range(1, nb_bins+1)) # range de 1 à 5
```

```
Nombre de clients par classe
----------------------------
```

	Class	R	F	M
0	1	18947	95179	19140
1	2	19147	221	19027
2	3	19112	9	19090
3	4	18995	8	19077
4	5	19217	1	19084

Concaténation des variables 'R', 'F', 'M'

```
                                 recency  frequency  monetary  R  F  M  RFM
customer_unique_id
0000366f3b9a7992bf8c76cfdf3221e2   116.0        1.0    141.90  4  1  4  414
0000b849f77a49e4a4ce2b2a4ca5be3f   119.0        1.0     27.19  4  1  1  411
0000f46a3911fa3c0805444483337064   542.0        1.0     86.22  1  1  3  113
0000f6ccb0745a6a4b88665a16c9f078   326.0        1.0     43.62  2  1  1  211
0004aac84e0df4da2b147fca70cf8255   293.0        1.0    196.89  2  1  4  214
```

- Nombre de valeurs uniques RFM

```
# Nombre de valeurs uniques RFM
rfm_count_unique = df_rfm.groupby('RFM')['RFM'].nunique()
print(rfm_count_unique.sum())
```

```
64
```

Segmentation RFM

Score RFM

- Somme des valeurs 'R' + 'F' + 'M'

customer_unique_id	recency	frequency	monetary	RFM	rfm_score
0000366f3b9a7992bf8c76cfdf3221e2	116.0	1.0	141.90	414	9
0000b849f77a49e4a4ce2b2a4ca5be3f	119.0	1.0	27.19	411	6
0000f46a3911fa3c0805444483337064	542.0	1.0	86.22	113	5
0000f6ccb0745a6a4b88665a16c9f078	326.0	1.0	43.62	211	4
0004aac84e0df4da2b147fca70cf8255	293.0	1.0	196.89	214	7

- Nombre de score RFM uniques

```
len(rfm_score_labels)
```

```
11
```

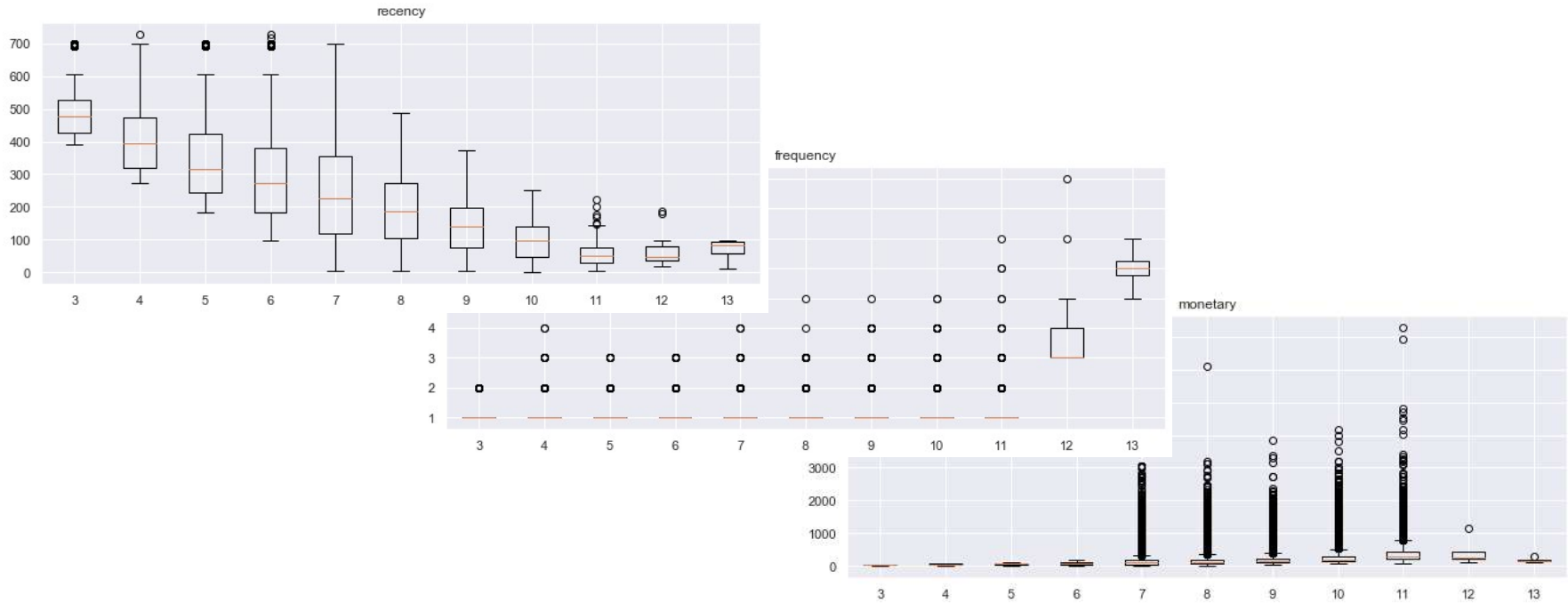
- Limites du modèle
  - Un même score RFM peut correspondre à des profils de clients très différents

6	114	1	1	4	ancien - occasionnel - grand montant
	123	1	2	3	
	141	1	4	1	ancien - récurrent - petit montant
	213	2	1	3	
	222	2	2	2	
	312	3	1	2	
	321	3	2	1	
	411	4	1	1	récent - occasionnel - petit montant

Stratégies commerciales

Option 1

- Elaboration des stratégies commerciales sur la base de 11 classes de clients et des statistiques de 'recency', 'frequency' et 'monetary' par classe



Stratégies commerciales

Option 1

- Elaboration des stratégies commerciales sur la base de 11 classes de clients et des statistiques de 'recency', 'frequency' et 'monetary' par classe

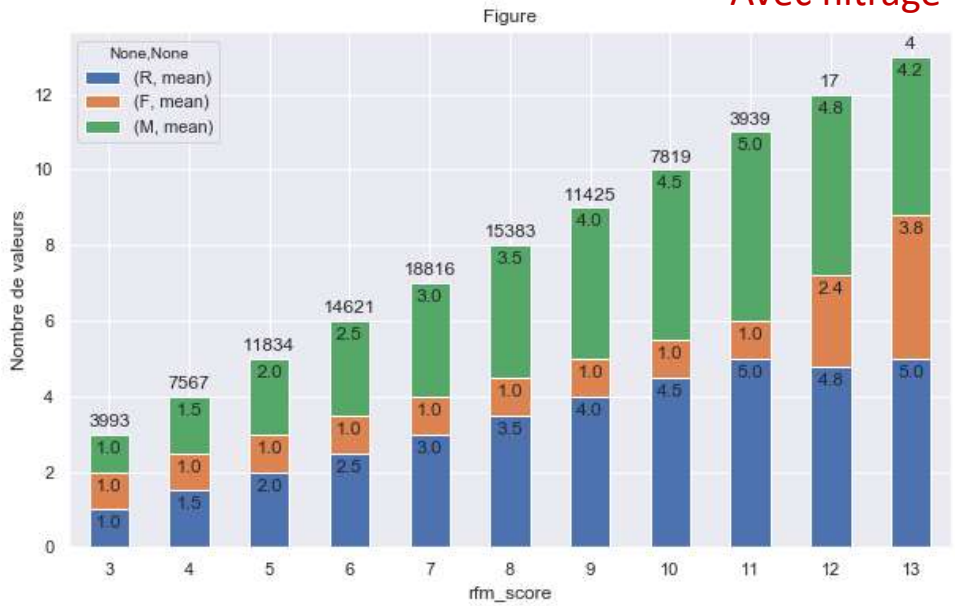
	duration				recency				frequency				monetary				
	mean	median	min	max	mean	median	min	max	mean	median	min	max	mean	median	min	max	count
rfm_score																	
3	1.3	1.0	1.0	167.0	482.3	476.0	390.0	700.0	1.0	1.0	1.0	2.0	39.6	40.0	12.3	55.0	3993
4	1.7	1.0	1.0	271.0	402.4	394.0	274.0	728.0	1.0	1.0	1.0	4.0	55.5	56.1	10.1	85.8	7567
5	2.1	1.0	1.0	348.0	339.1	316.0	183.0	701.0	1.0	1.0	1.0	3.0	71.1	68.1	13.8	129.1	11834
6	2.4	1.0	1.0	399.0	289.5	272.0	99.0	729.0	1.0	1.0	1.0	6.0	92.7	85.1	12.9	202.5	14621
7	3.4	1.0	1.0	484.0	245.3	227.0	5.0	699.0	1.0	1.0	1.0	4.0	162.7	106.2	9.6	6929.3	18816
8	4.8	1.0	1.0	634.0	186.3	186.0	5.0	489.0	1.0	1.0	1.0	5.0	194.7	131.6	29.4	6081.5	15383
9	5.1	1.0	1.0	610.0	138.2	140.0	5.0	375.0	1.0	1.0	1.0	5.0	217.9	158.9	58.5	3826.8	11425
10	5.7	1.0	1.0	574.0	95.6	97.0	0.0	251.0	1.0	1.0	1.0	5.0	296.5	200.4	88.4	4175.3	7819
11	6.9	1.0	1.0	538.0	52.8	49.0	5.0	222.0	1.0	1.0	1.0	7.0	433.1	301.8	90.1	7274.9	3939
12	220.3	209.0	2.0	483.0	65.9	47.0	19.0	188.0	4.0	3.0	3.0	9.0	342.1	260.9	108.4	1164.0	17
13	320.2	313.0	236.0	419.0	69.2	84.0	11.0	98.0	6.0	6.0	5.0	7.0	197.1	166.3	137.7	318.2	4

Stratégies commerciales

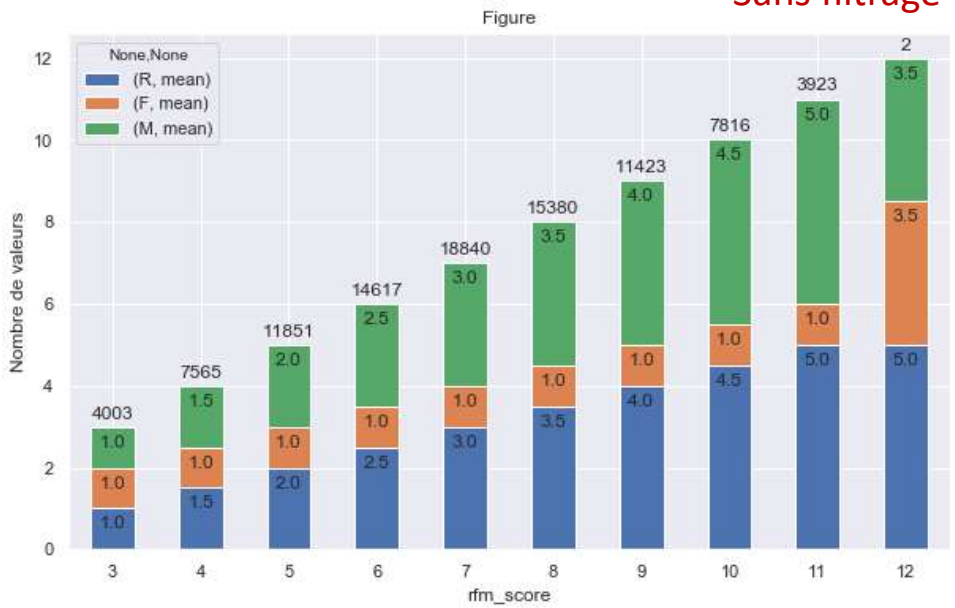
Option 2

- Réduire le nombre de classes de clients en fonction de contraintes économique et logistique de l'entreprise
- On produit des statistiques 'R', 'F', 'M' plus faciles à analyser

Avec filtrage



Sans filtrage

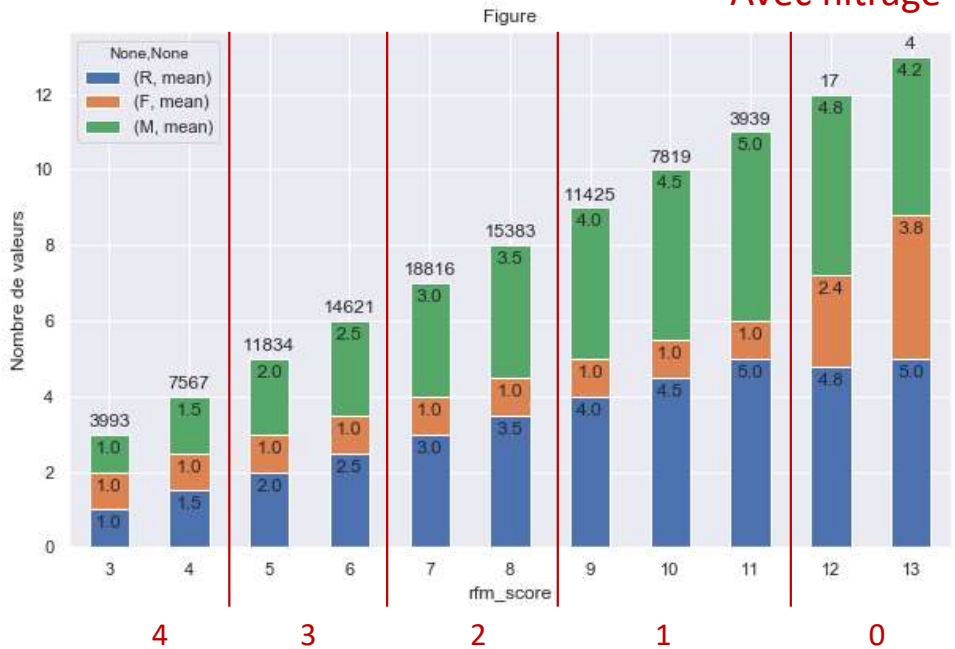


Stratégies commerciales

Option 2

- Réduire le nombre de classes de clients en fonction de contraintes économique et logistique de l'entreprise
- On choisit 5 classes

Avec filtrage



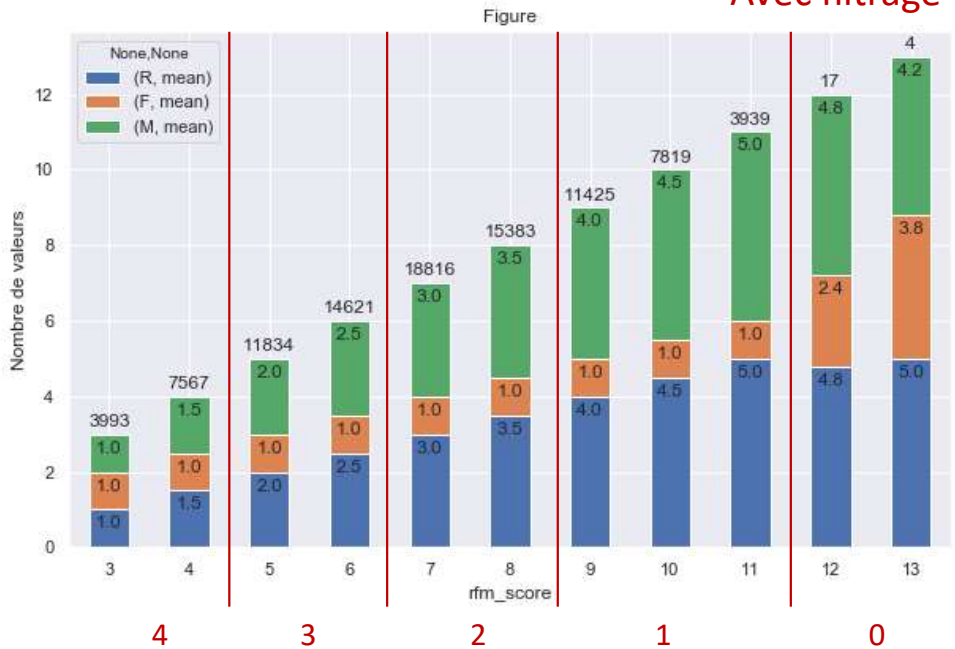

```python
list_segment = []
list_segment_name = []

for i in df_rfm['rfm_score'].values.tolist():
        if (i >= 12):
            list_segment_name.append('Très récent - Récurrent - Grand CA')
            list_segment.append(0)
        elif (i >= 9) and (i <= 11):
            list_segment_name.append('Très récent - Occasionnel - Grand CA')
            list_segment.append(1)
        elif (i >= 7) and (i <= 8):
            list_segment_name.append('Récent - Occasionnel - Moyen CA')
            list_segment.append(2)
        elif (i >= 5) and (i <= 6):
            list_segment_name.append('Ancien - Occasionnel - Moyen CA')
            list_segment.append(3)
        else:
            list_segment_name.append('Ancien - Occasionnel - Petit CA')
            list_segment.append(4)
```

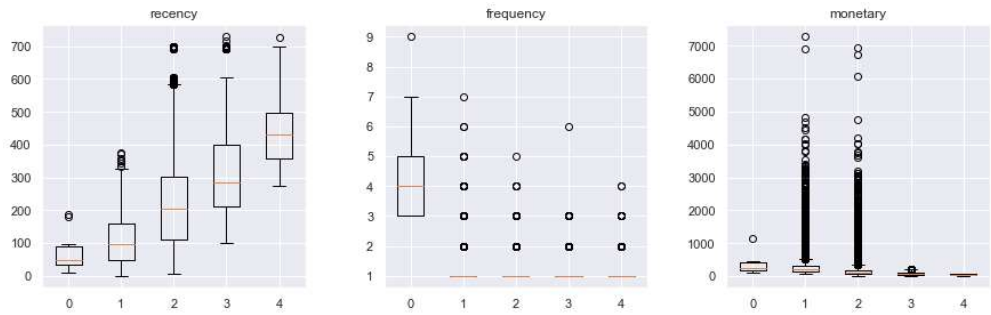
Stratégies commerciales

Option 2

- Réduire le nombre de classes de clients en fonction de contraintes économique et logistique de l'entreprise
- On choisit 5 classes

Avec filtrage





[0] Très récent - Récurrent - Grand CA (21)

[1] Très récent - Occasionnel - Grand CA (23183)

[2] Récent - Occasionnel - Moyen CA (34199)

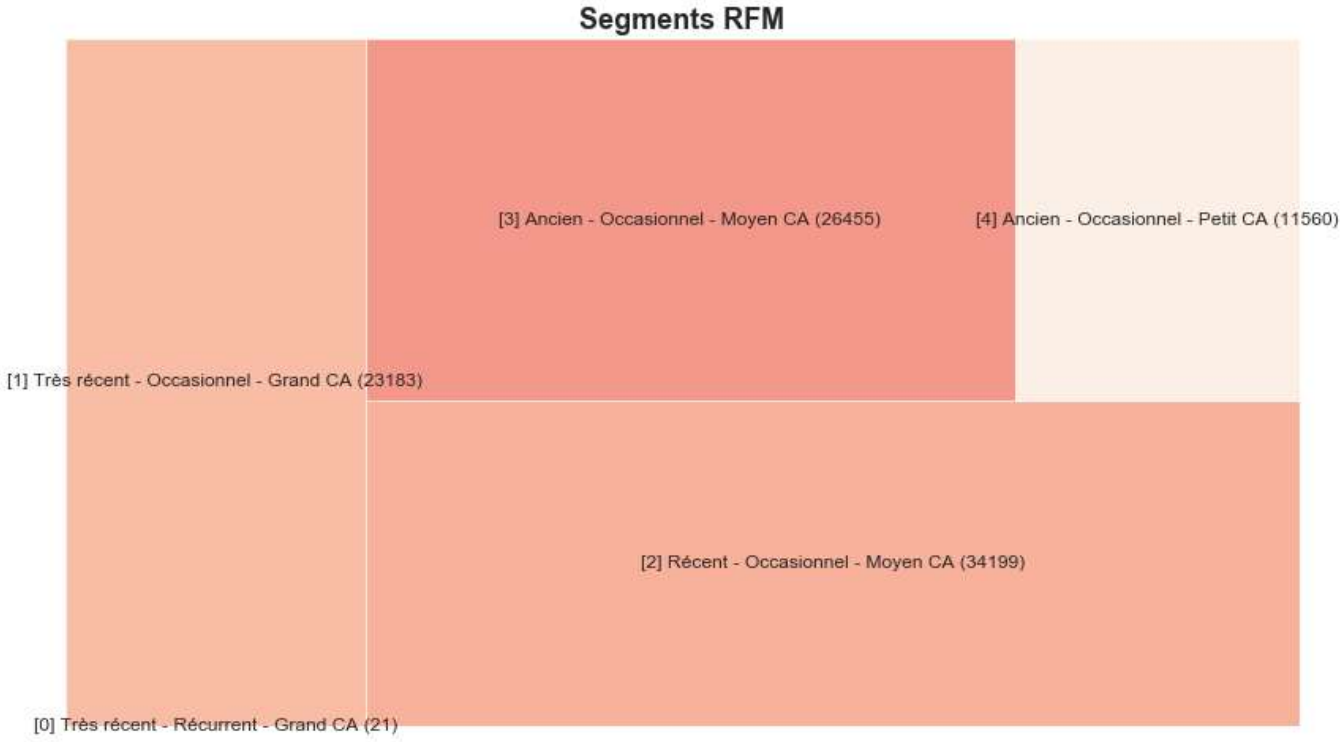
[3] Ancien - Occasionnel - Moyen CA (26455)

[4] Ancien - Occasionnel - Petit CA (11560)

Stratégies commerciales

Option 2

- Représentation proportionnelle des clients par segment



Regrouper des individus ayant des caractéristiques similaires

Clustering

Le **partitionnement de données** (ou ***data clustering*** en anglais) est une méthode en analyse des données.

Elle vise à diviser un ensemble de données en différents « paquets » homogènes,

en ce sens que les données de chaque sous-ensemble partagent des caractéristiques communes,

qui correspondent le plus souvent à des critères de proximité (similarité informatique) que l'on définit en introduisant des mesures et classes de distance entre objets.

Pour obtenir un bon partitionnement, il convient d'à la fois :

→ minimiser l'inertie *intra-classe* pour obtenir des grappes (*cluster* en anglais) les plus homogènes possibles

→ maximiser l'inertie inter-classe afin d'obtenir des sous-ensembles bien différenciés.

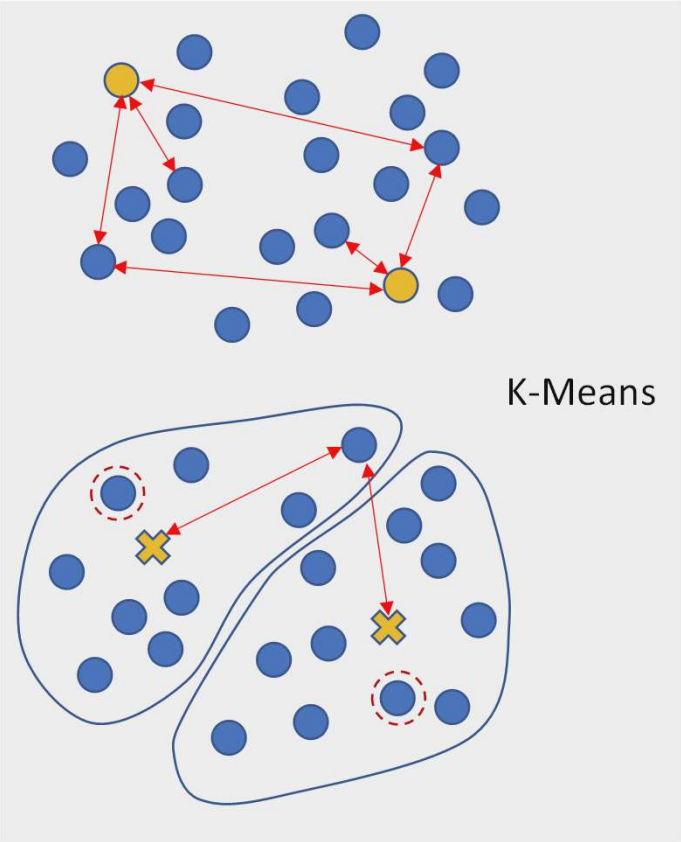
Source: https://fr.wikipedia.org/wiki/Partitionnement_de_donn%C3%A9es

Algorithmes

K-Means

- On tire aléatoirement n individus (n = nombre de clusters)

1) Calcul des distances entre chaque individu et les centres initiaux
2) Regroupement des individus selon le centre le proche
3) Redéfinition des centres
4) On réitère les opérations 1 à 3

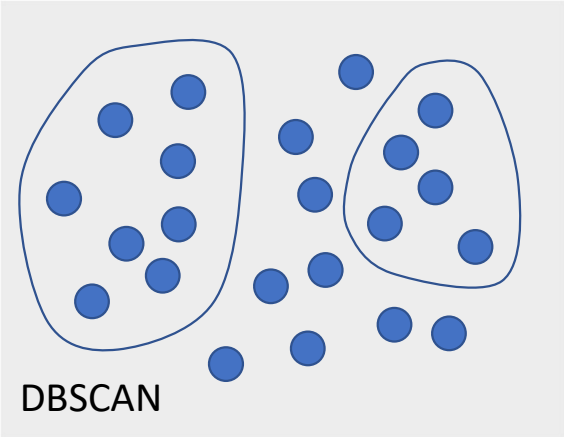


K-Prototypes

- Variables numériques et catégorielles

DBSCAN

- Densité



AffinityPropagation

- Test non concluant

Jeu de données

Dataframe

```
df_rfm.columns.tolist()
```

```
['customer_unique_id',
 'start',
 'end',
 'older',
 'recency',
 'duration',
 'frequency',
 'frequency_2',
 'prod_nb_avg',
 'nb_prod_diff',
 'nb_pay_type_diff',
 'monetary',
 'price_mean',
 'price_total',
 'same_loc',
 'score',
 'R',
 'F',
 'M',
 'RFM',
 'rfm_score',
 'segment_x',
 'segment_y']
```

Variables

```
df_X.head(2)
```

customer_unique_id	recency	frequency	monetary	price_total
0000366f3b9a7992bf8c76cfdf3221e2	116.0	1.0	141.90	141.90
0000b849f77a49e4a4ce2b2a4ca5be3f	119.0	1.0	27.19	27.19

```
df_X.shape
```

```
(95418, 4)
```

Standardisation des données

```
from sklearn import preprocessing
X_std = preprocessing.scale(df_X)
```

```
X_std[:2]
```

```
array([[-0.83248397, -0.16469323, -0.08764627, -0.10706047],
       [-0.81289655, -0.16469323, -0.6166779 , -0.61895693]])
```

Analyse en Composantes Principales

Définition

L'ACP est utilisée pour décomposer un jeu de données multivarié en un ensemble de composantes orthogonales successives qui expliquent une quantité maximale de la variance.

→ Moins de variables, maximum de variance

Objectifs

- Optimisation K-Means
  - A priori, valable pour les jeux de grandes dimensions
  - Réduction dimensionnelle
- Visualisation des données
  - Représentation 2D

```
from sklearn import decomposition

pca = decomposition.PCA(n_components = len(list_var)-1)
X = pca.fit_transform(X_std)

print(pca.explained_variance_ratio_.cumsum())
```

```
[0.34139797 0.67472219 1.        ]
```

On rajoute les données catégorielles au jeu de données standardisé et transformé par ACP.

```
df_X_pc = pd.DataFrame(X)

df_X_c_1 = df_X_pc.merge(df_cat_1, left_index=True, right_index=True)
del df_X_c_1['customer_unique_id']
X_c_1 = np.array(df_X_c_1)
```

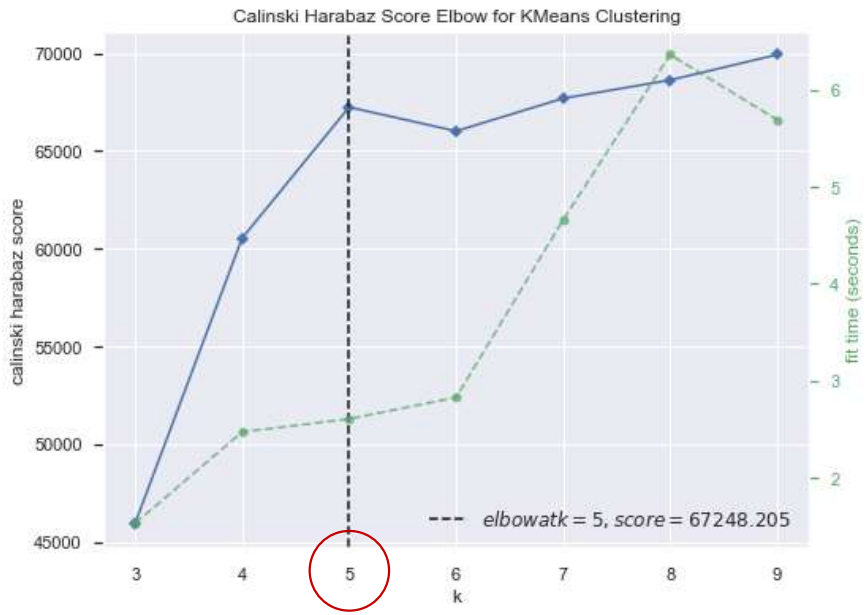
K-Means

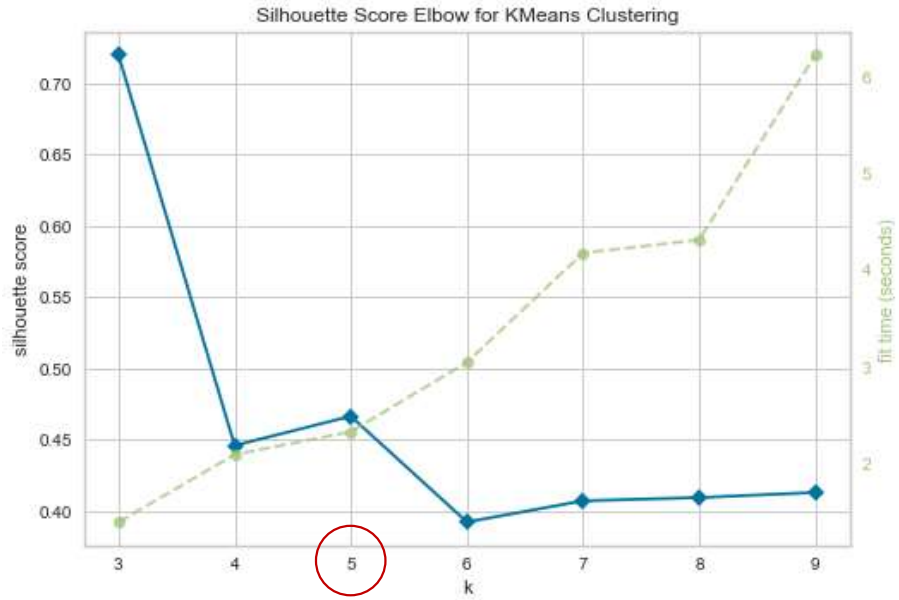
Détermination du nombre optimal de clusters

- Variance inter-groupes et intra-groupe
  - Rapport entre la variance inter-groupes et la variance intra-groupe
  - Maximisation du score

```
for num_clusters in range(3, 10):

    kmea = cluster.KMeans(n_clusters = num_clusters,
                          init = 'k-means++',
                          n_init = 10,
                          max_iter = 300,
                          tol = 1e-04,
                          precompute_distances = 'auto',
                          algorithm = 'auto',
                          random_state = 42,
                          verbose = 0)

    kmea.fit(X)
```

- Coefficient de silhouette

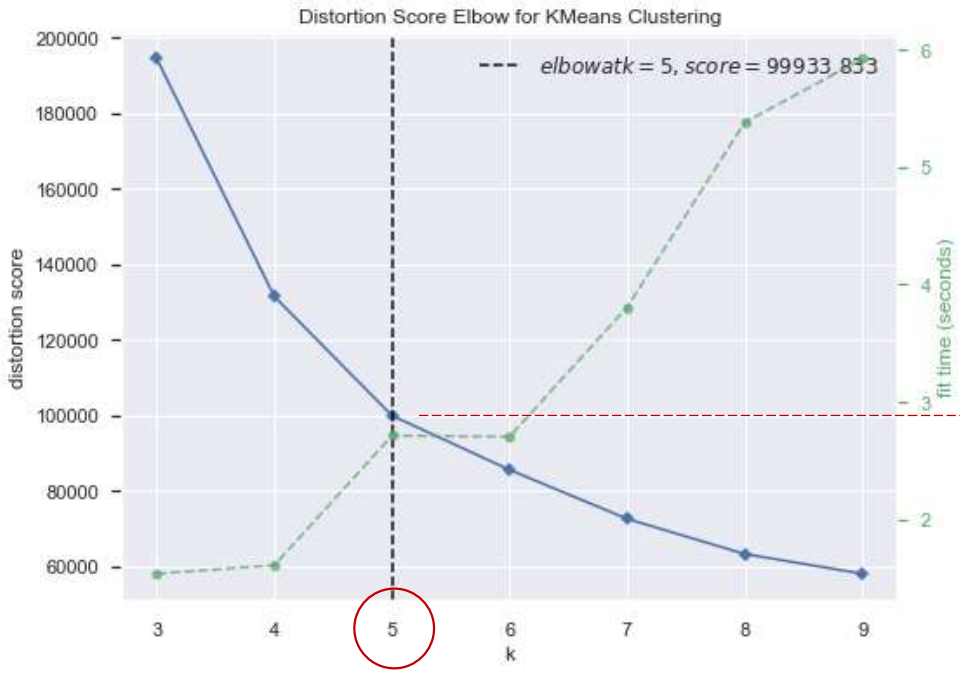




K-Means

Détermination du nombre optimal de clusters

- Distorsion ou Inertie
  - Somme des distances au carré de chaque point aux centroïdes



Paramétrisation

```
{'algorithm': 'auto',
 'copy_x': True,
 'init': 'k-means++',
 'max_iter': 300,
 'n_clusters': 5,
 'n_init': 10,
 'n_jobs': None,
 'precompute_distances': 'auto',
 'random_state': 42,
 'tol': 0.0001,
 'verbose': 0}

Inertie: 99933.905216514
```

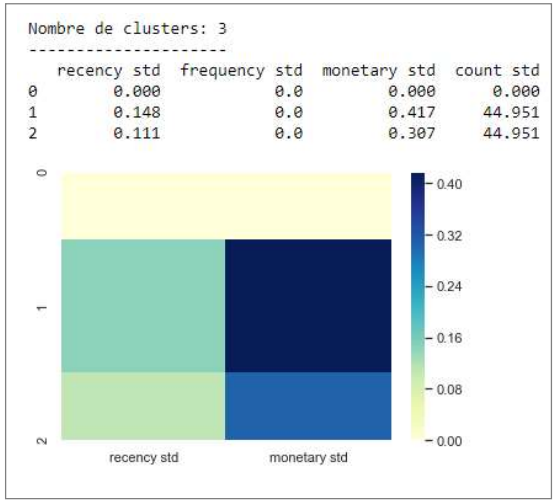
K-Means

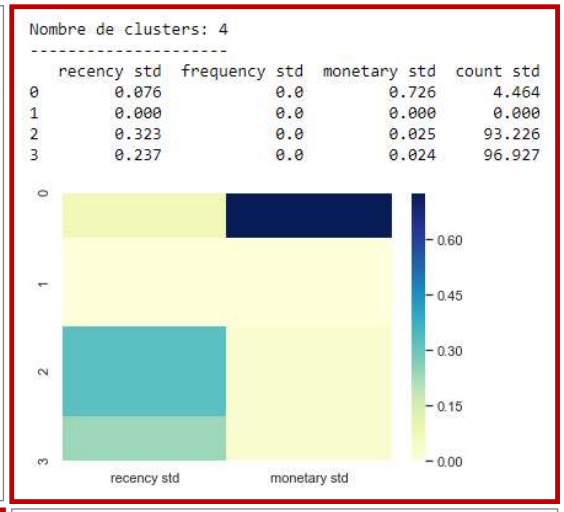
Stabilité

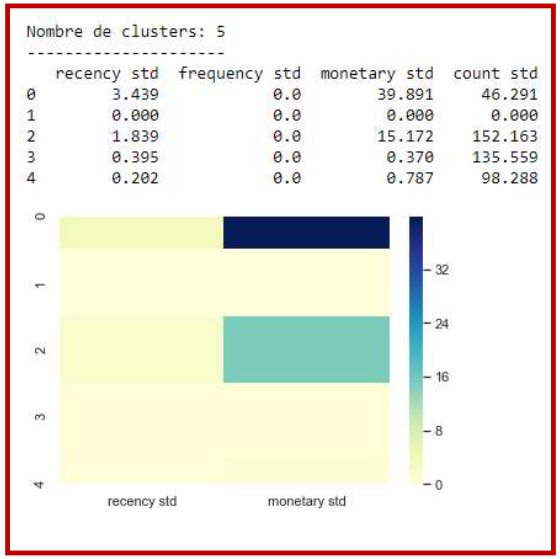
- On exécute l'algorithme K-Means n fois avec des random states différents
- On stocke les moyennes des variables par groupe

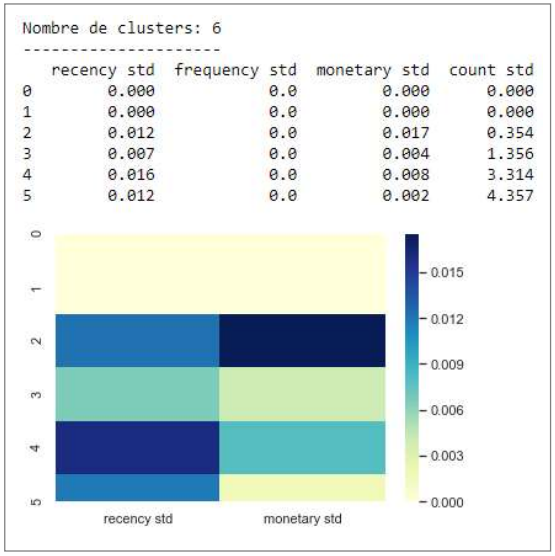
```
Statistiques (pour le dernier random state testé)
-------------------------------------------------

              recency frequency     monetary
                 mean      mean         mean  count
KM_group
0           250.282971  1.026889  1790.130935    781
2           226.054615  2.109575   139.442028   2893
1           236.155595  1.000000   598.665690   5585
3           464.536380  1.000000   121.329534  20616
4           260.151571  1.000000   114.054115  32493
5            91.866657  1.000000   121.054026  33050
```

- On mesure la variation des valeurs de chaque variable pour les n exécutions
- Méthode de mesure:
  - Ecart-type









K-Means

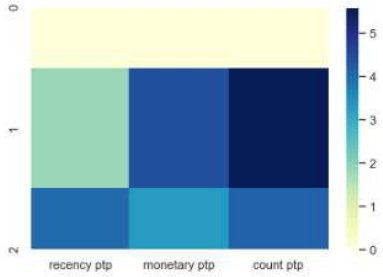
Stabilité

- On exécute l'algorithme K-Means n fois avec des random states différents
- On stocke les moyennes des variables par groupe

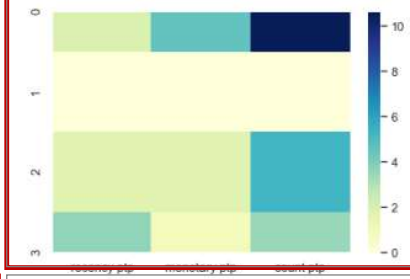
```
Statistiques (pour le dernier random state testé)
-------------------------------------------------

             recency frequency     monetary
                mean      mean         mean  count
KM_group
0          250.282971  1.026889  1790.130935    781
2          226.054615  2.109575   139.442028   2893
1          236.155595  1.000000   598.665690   5585
3          464.536380  1.000000   121.329534  20616
4          260.151571  1.000000   114.054115  32493
5           91.866657  1.000000   121.054026  33050
```

- On mesure la variation des valeurs de chaque variable pour les n exécutions
- Méthode de mesure:
  - Amplitude

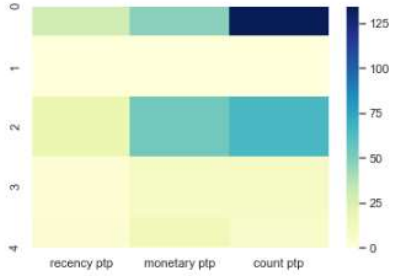
```
Nombre de clusters: 3
---------------------

Taux de dispersions pour l'ensemble des variables
-------------------------------------------------

   recency ptp  frequency ptp  monetary ptp  count ptp
0        0.000            0.0         0.000      0.000
1        1.845            0.0         4.422      5.574
2        4.007            0.0         3.298      4.115
```



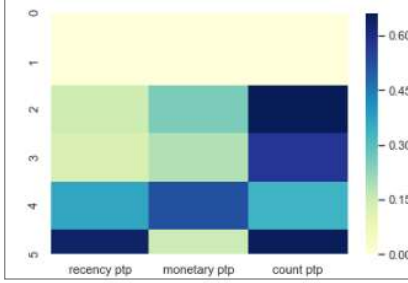

```
Nombre de clusters: 4
---------------------

Taux de dispersions pour l'ensemble des variables
-------------------------------------------------

   recency ptp  frequency ptp  monetary ptp  count ptp
0        1.965          0.022         4.527     10.603
1        0.000          0.000         0.000      0.000
2        1.710          0.000         1.780      5.301
3        3.679          0.000         0.969      3.516
```



```
Nombre de clusters: 5
---------------------

Taux de dispersions pour l'ensemble des variables
-------------------------------------------------

   recency ptp  frequency ptp  monetary ptp  count ptp
0       28.887          0.996        47.933    134.326
1        0.000          0.000         0.000      0.000
2       17.636          0.000        53.941     64.977
3        2.337          0.000         7.354      8.783
4        4.070          0.000        14.643      6.367
```




```
Nombre de clusters: 6
---------------------

Taux de dispersions pour l'ensemble des variables
-------------------------------------------------

   recency ptp  frequency ptp  monetary ptp  count ptp
0        0.000            0.0         0.000      0.000
1        0.000            0.0         0.000      0.000
2        0.149            0.0         0.253      0.662
3        0.125            0.0         0.191      0.574
4        0.368            0.0         0.524      0.334
5        0.640            0.0         0.153      0.657
```



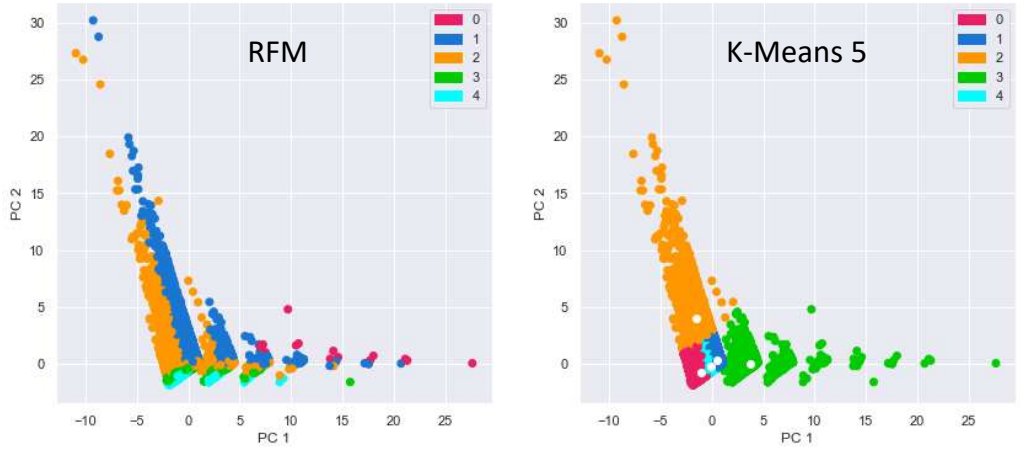
K-Means

Visualisation des groupes (RFM vs K-Means 5)

segment	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0.0	#E91E63	fuchsia	66.57	4.38	314.50	21
1.0	#1976D2	bleu	109.33	1.04	280.99	23183
2.0	#FF9800	orange	218.78	1.03	177.12	34199
3.0	#00CC00	vert	311.70	1.03	83.04	26455
4.0	#00FFFF	bleu (aqua)	429.99	1.02	50.01	11560

KM_group_5	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	464.24	1.00	132.99	21311
1	#1976D2	bleu	92.24	1.00	137.16	34573
2	#FF9800	orange	247.75	1.00	1130.02	2576
3	#00CC00	vert	225.87	2.11	143.15	2906
4	#00FFFF	bleu (aqua)	260.11	1.00	130.69	34052

```
data_c_plot(segment_l,kmea5_l,0,kmea5)
```



K-Means

Statistiques

- RFM

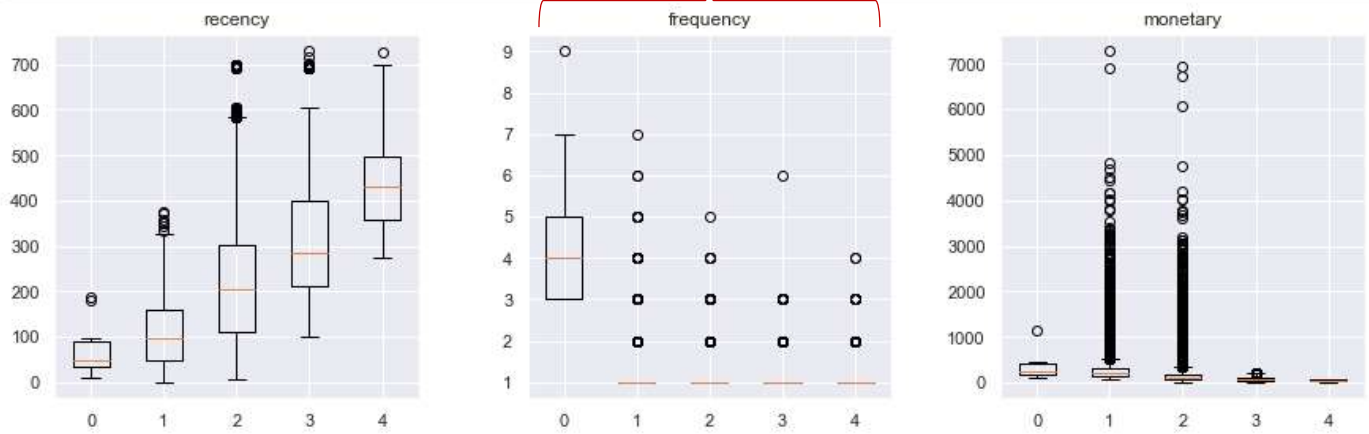
Segment	Nombre de clients
0.0	21
1.0	23183
2.0	34199
3.0	26455
4.0	11560

- K-Means 5

KM_group_5	Nombre de clients
0	21311
1	34573
2	2576
3	2906
4	34052

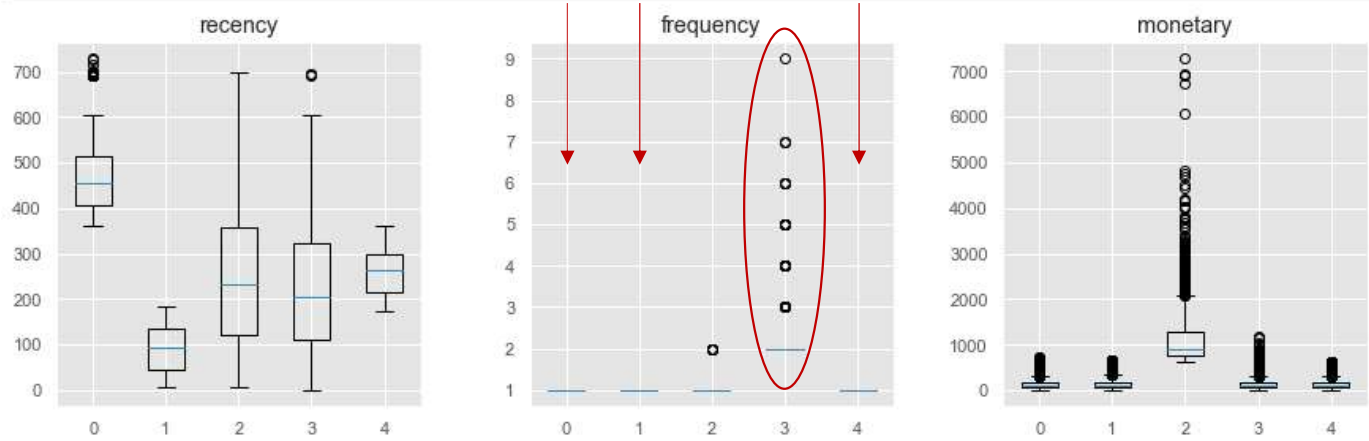
Chaque groupe contient des client
avec plusieurs commandes

```
data_boxplot(df_rfm,segment,1)
```




```
data_boxplot(df_rfm,'KM_group_5',1)
```

1 commande par client



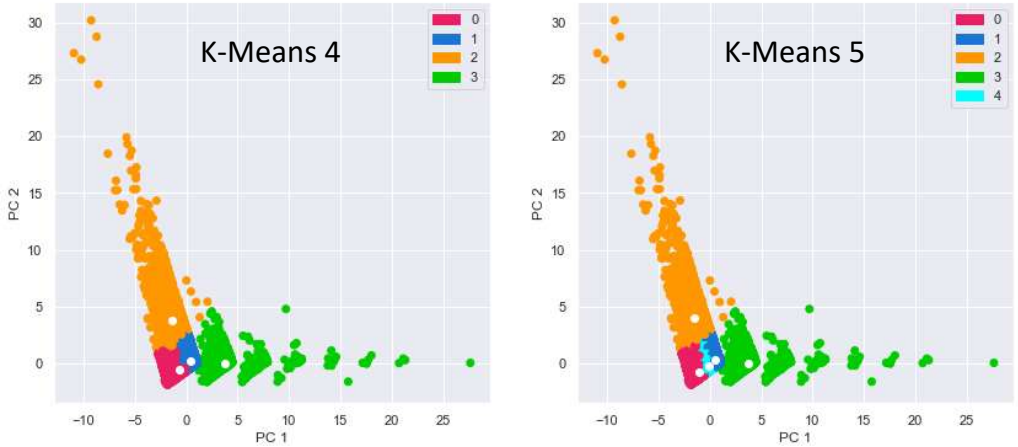
K-Means

Visualisation des groupes (K-Means 4 vs K-Means 5)

KM_group_4	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	393.97	1.00	131.51	38030
1	#1976D2	bleu	133.47	1.00	132.78	51639
2	#FF9800	orange	247.23	1.00	1083.12	2843
3	#00CC00	vert	225.87	2.11	143.15	2906

KM_group_5	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	464.24	1.00	132.99	21311
1	#1976D2	bleu	92.24	1.00	137.16	34573
2	#FF9800	orange	247.75	1.00	1130.02	2576
3	#00CC00	vert	225.87	2.11	143.15	2906
4	#00FFFF	bleu (aqua)	260.11	1.00	130.69	34052

```
data_c_plot(kmea4_l,kmea5_l,kmea4,kmea5)
```



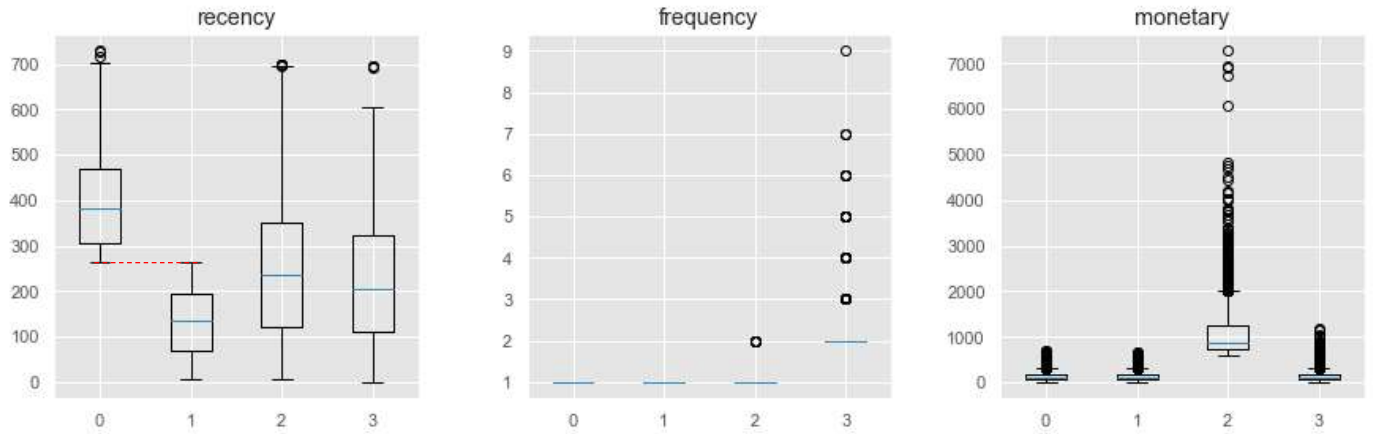
K-Means

Statistiques

- K-Means 4

KM_group_4	Nombre de clients
0	38030
1	51639
2	2843
3	2906

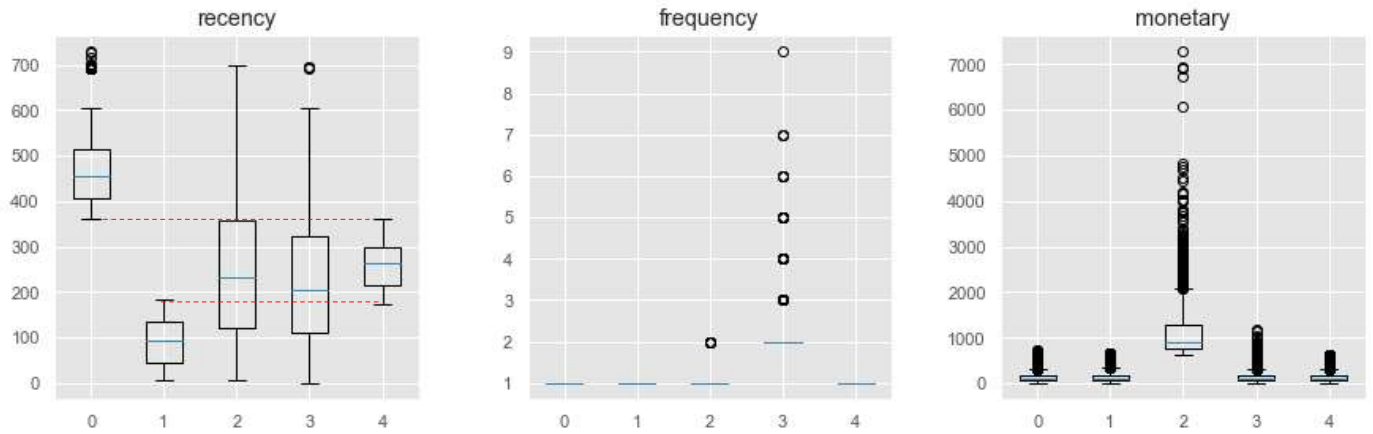
```
data_boxplot(df_rfm,'KM_group_4',1)
```



- K-Means 5

KM_group_5	Nombre de clients
0	21311
1	34573
2	2576
3	2906
4	34052

```
data_boxplot(df_rfm,'KM_group_5',1)
```



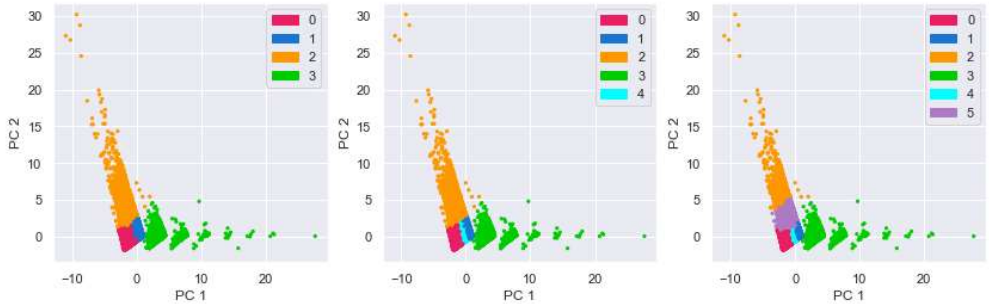
K-Means

Revenons à…

… la stabilité

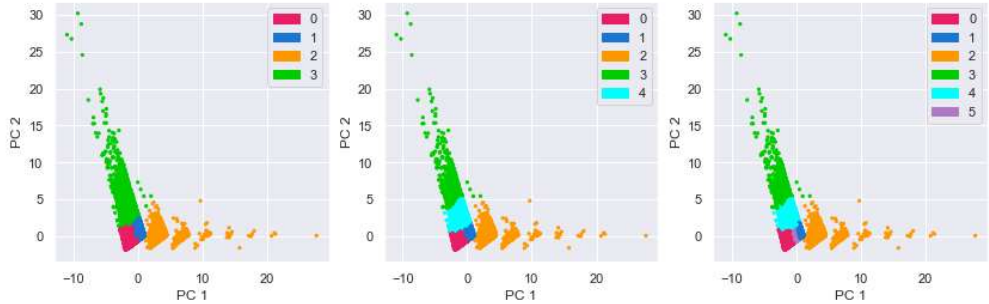
… Quid du pattern 2 ?

Random state: 42



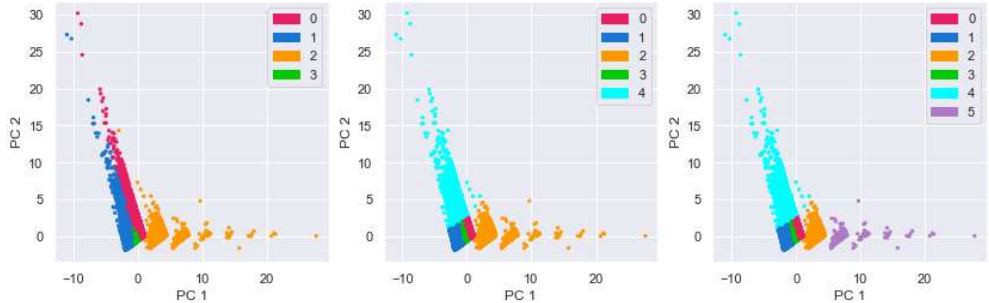
pattern 1

Random state: 39



pattern 2

Random state: 36



pattern 3

```
Durée de l'opération 'Stabilité (2) - k-means': 65.31 s
```

K-Means

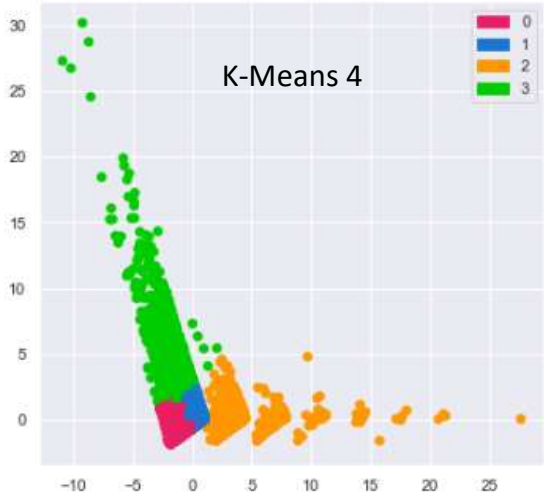
Visualisation des groupes (K-Means 4 vs K-Means 5 pattern 2)

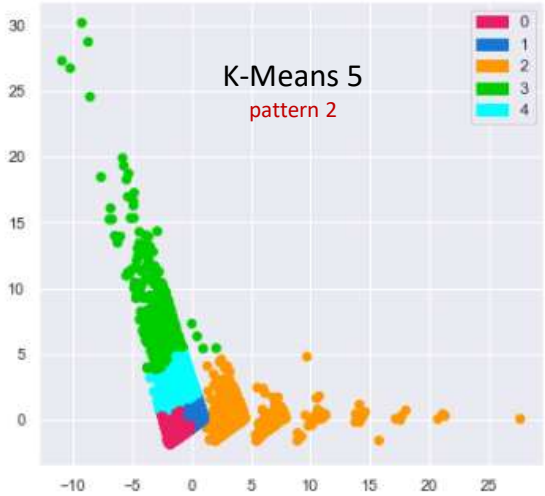
KM_group_4	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	394.43	1.00	131.84	37921
1	#1976D2	bleu	133.85	1.00	133.40	51840
2	#FF9800	orange	225.87	2.11	143.15	2906
3	#00CC00	vert	247.98	1.00	1098.54	2751

KM_group_5	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	396.41	1.00	121.99	36601
1	#1976D2	bleu	134.24	1.00	120.53	50369
2	#FF9800	orange	225.92	2.11	143.52	2907
3	#00CC00	vert	245.10	1.01	1893.93	663
4	#00FFFF	bleu (aqua)	234.66	1.00	644.58	4878

```
random_state =  39
```

```
data_c_plot(kmea4_l, kmea5_l)
```





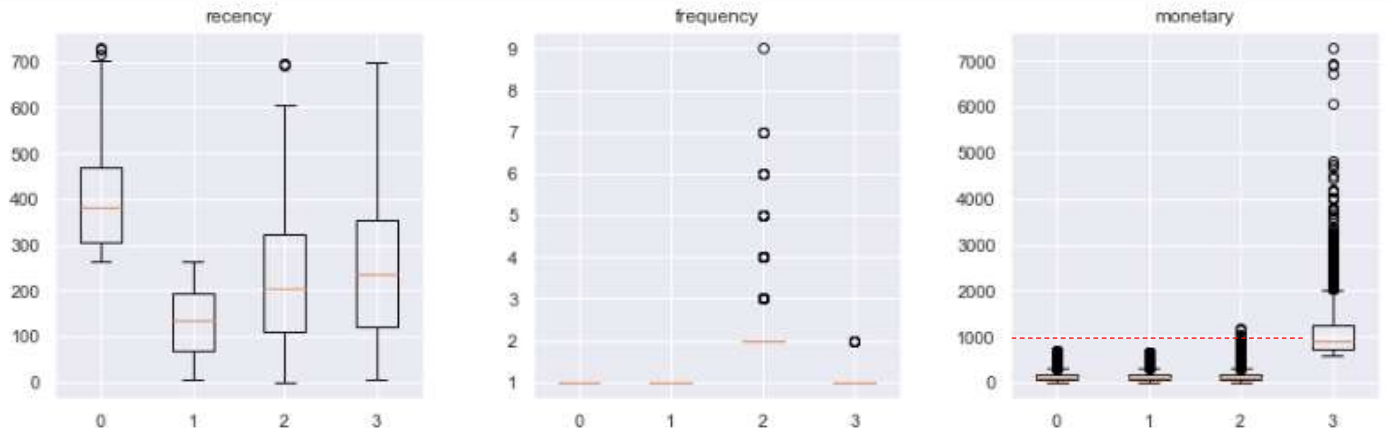
K-Means

Statistiques

- K-Means 4

KM_group_4	Nombre de clients
0	37921
1	51840
2	2906
3	2751

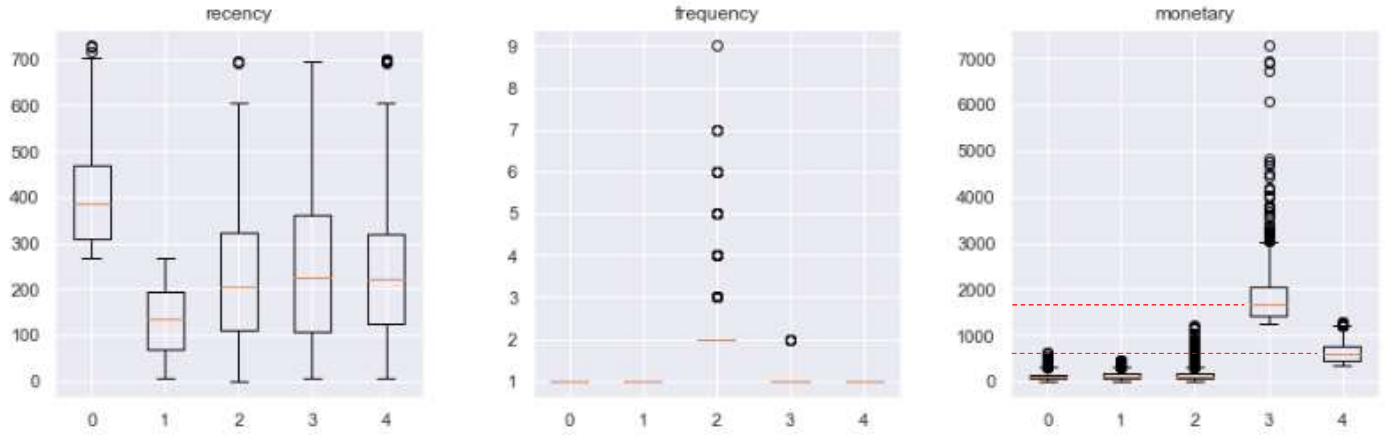
```
data_boxplot(df_rfm,'KM_group_4',1)
```



- K-Means 5 pattern 2

KM_group_5	Nombre de clients
0	36601
1	50369
2	2907
3	663
4	4878

```
data_boxplot(df_rfm,'KM_group_5',1)
```



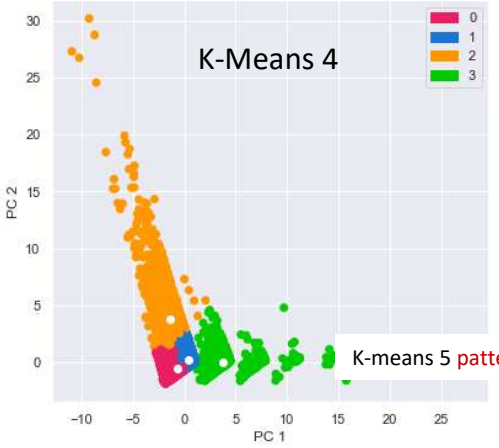
K-Means

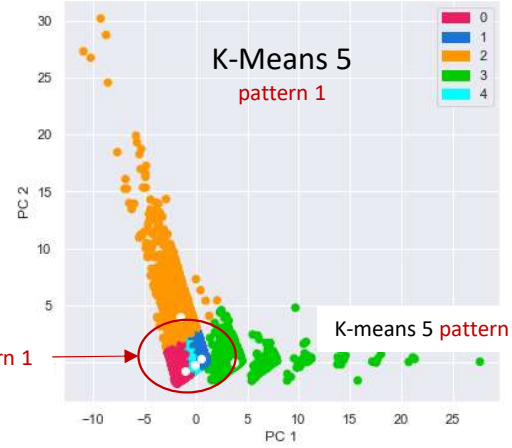
Visualisation des groupes (K-Means 6)

KM_group_6	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	464.81	1.00	125.80	20889
1	#1976D2	bleu	91.92	1.00	124.15	33499
2	#FF9800	orange	245.53	1.01	1902.43	654
3	#00CC00	vert	225.92	2.11	143.52	2907
4	#00FFFF	bleu (aqua)	260.36	1.00	118.24	32932
5	#6C3483	violet	232.45	1.00	663.70	4537

KM_group_5	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	396.41	1.00	121.99	36601
1	#1976D2	bleu	134.24	1.00	120.53	50369
2	#FF9800	orange	225.92	2.11	143.52	2907
3	#00CC00	vert	245.10	1.01	1893.93	663
4	#00FFFF	bleu (aqua)	234.66	1.00	644.58	4878

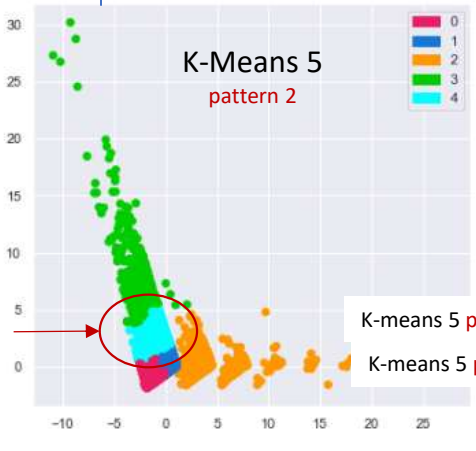
```
data_c_plot(kmea6_l, kmea5_l)
```

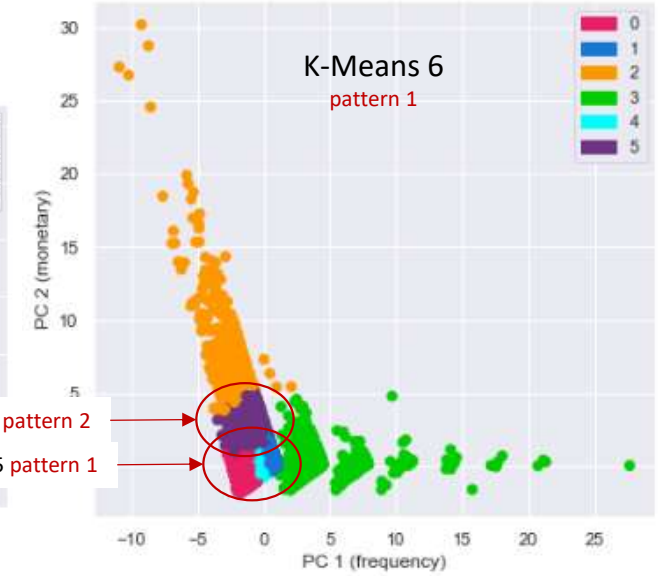




K-means 5 pattern 1

K-means 5 pattern 2





K-means 5 pattern 2

K-means 5 pattern 1

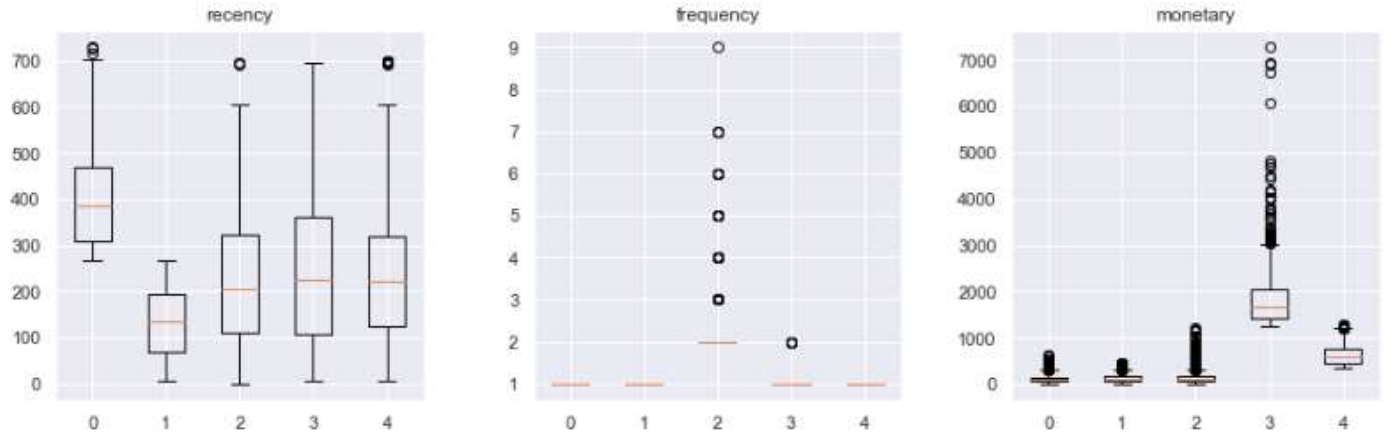
K-Means

Statistiques

- K-Means 5 pattern 2

KM_group_5	Nombre de clients
0	36601
1	50369
2	2907
3	663
4	4878

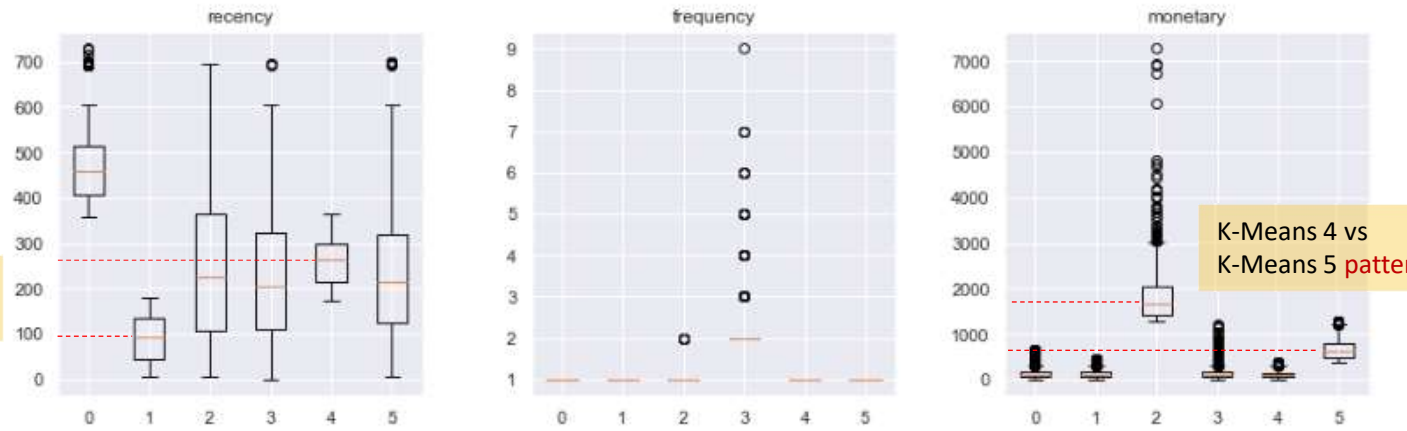
```
data_boxplot(df_rfm,'KM_group_5',1)
```



- K-Means 6

KM_group_6	Nombre de clients
0	20889
1	33499
2	654
3	2907
4	32932
5	4537

```
data_boxplot(df_rfm,'KM_group_6',1)
```



K-Means 4 vs
K-Means 5

K-Means 4 vs
K-Means 5 pattern 2

Nouvelles variables (1)

	recency	frequency	monetary	price_total
customer_unique_id				
0000366f3b9a7992bf8c76cfdf3221e2	116.0	1.0	141.9	141.9

Visualisation des groupes (K-Means 5)

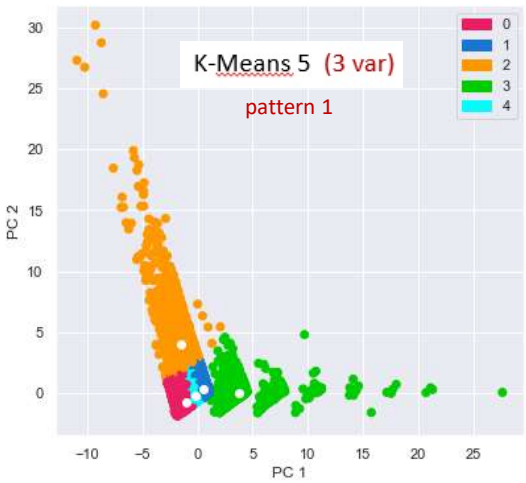
KM_group_5	c_values	c_names	(recency, mean)	(frequency, mean)	(monetary, mean)	Nombre de clients
0	#E91E63	fuchsia	464.24	1.00	132.99	21311
1	#1976D2	bleu	92.24	1.00	137.16	34573
2	#FF9800	orange	247.75	1.00	1130.02	2576
3	#00CC00	vert	225.87	2.11	143.15	2906
4	#00FFFF	bleu (aqua)	260.11	1.00	130.69	34052

KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	396.12	1.00	117.99	36144
1	#1976D2	bleu	134.29	1.00	117.31	49721
2	#FF9800	orange	226.05	2.11	139.44	2893
3	#00CC00	vert	250.41	1.03	1784.87	788
4	#00FFFF	bleu (aqua)	236.55	1.00	586.79	5872

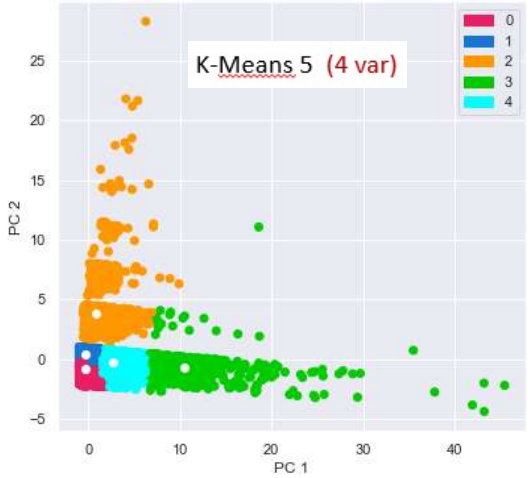
```
print(pca.explained_variance_ratio_.cumsum())
```

```
[0.34139797 0.67472219 1.        ]
```

```
data_c_plot(kmea5_l,kmea6_l,kmea5,kmea6)
```




```
data_c_plot(kmea5_l,kmea6_l,kmea5,kmea6)
```




```
print(pca.explained_variance_ratio_.cumsum())
```

```
[0.49712866 0.75288955 0.99777527 1.        ]
```

recency	frequency	monetary

Nouvelles variables (1)

Statistiques

- K-Means 5 (3 var)

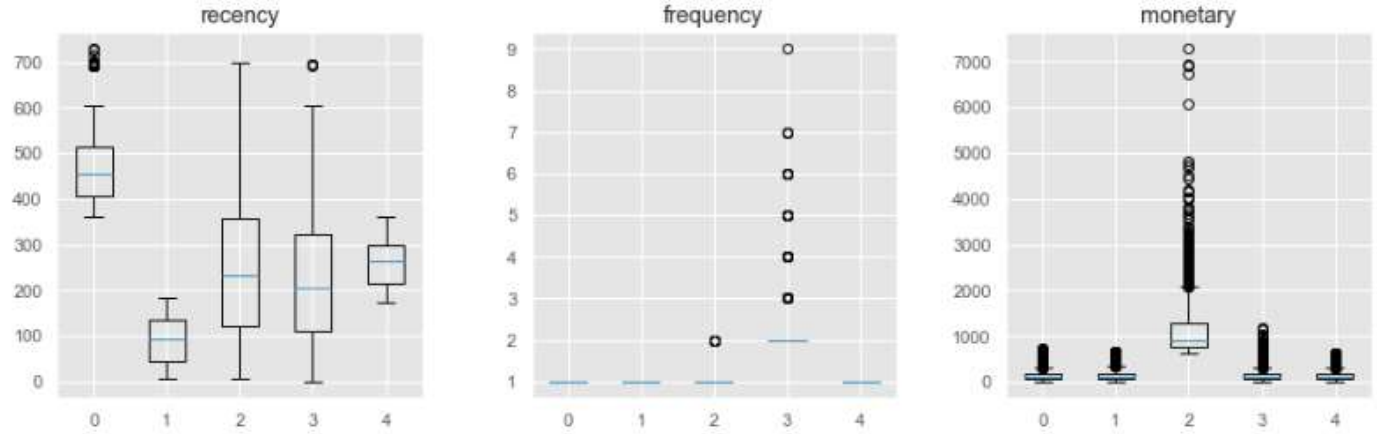
KM_group_5	Nombre de clients
0	21311
1	34573
2	2576
3	2906
4	34052

- K-Means 5 (4 var)

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

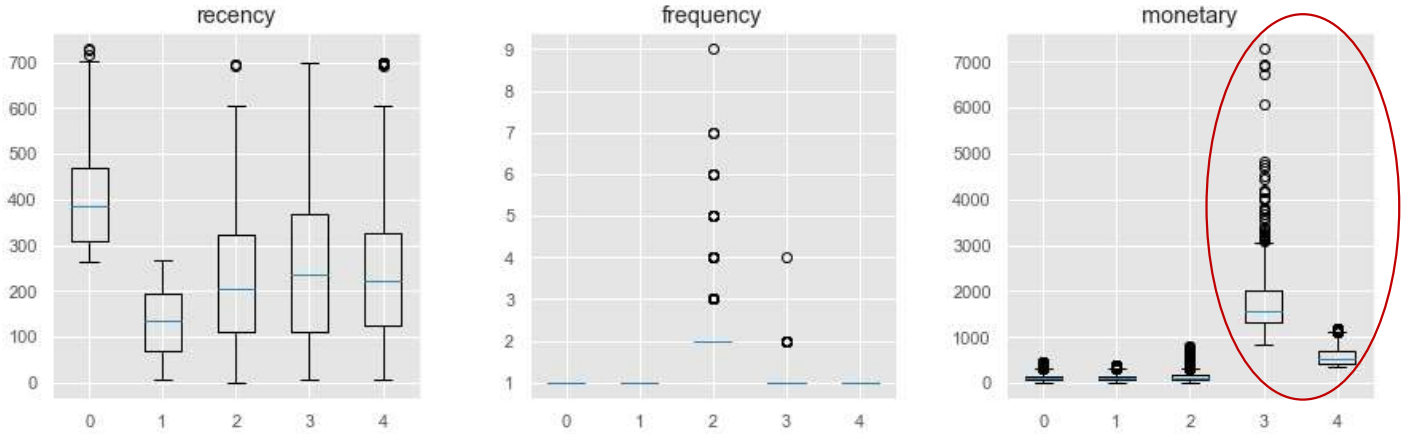
```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary




```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total



Nouvelles variables (1)

Statistiques

- K-Means 5 (3 var)

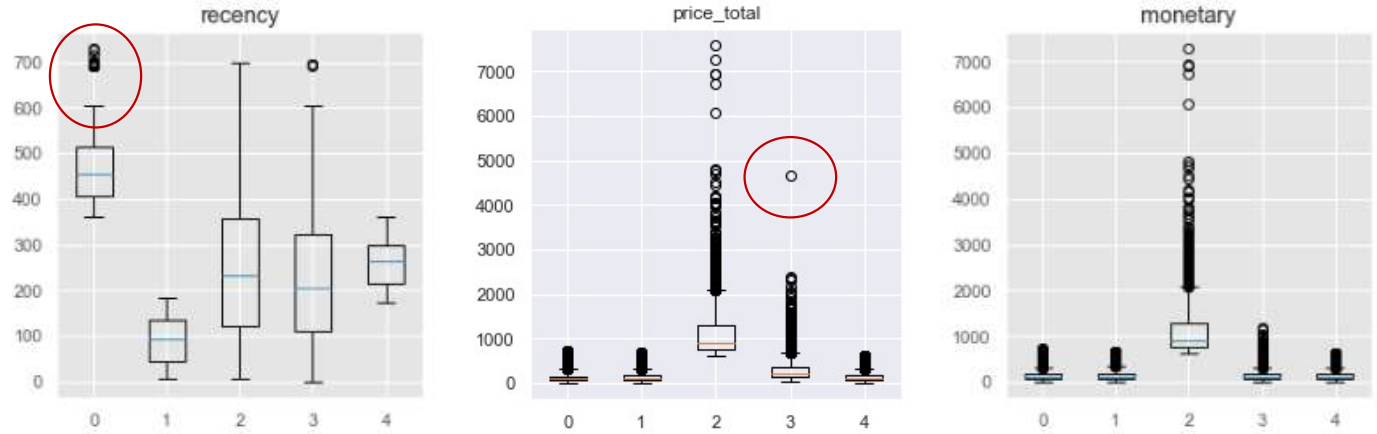
KM_group_5	Nombre de clients
0	21311
1	34573
2	2576
3	2906
4	34052

- K-Means 5 (4 var)

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

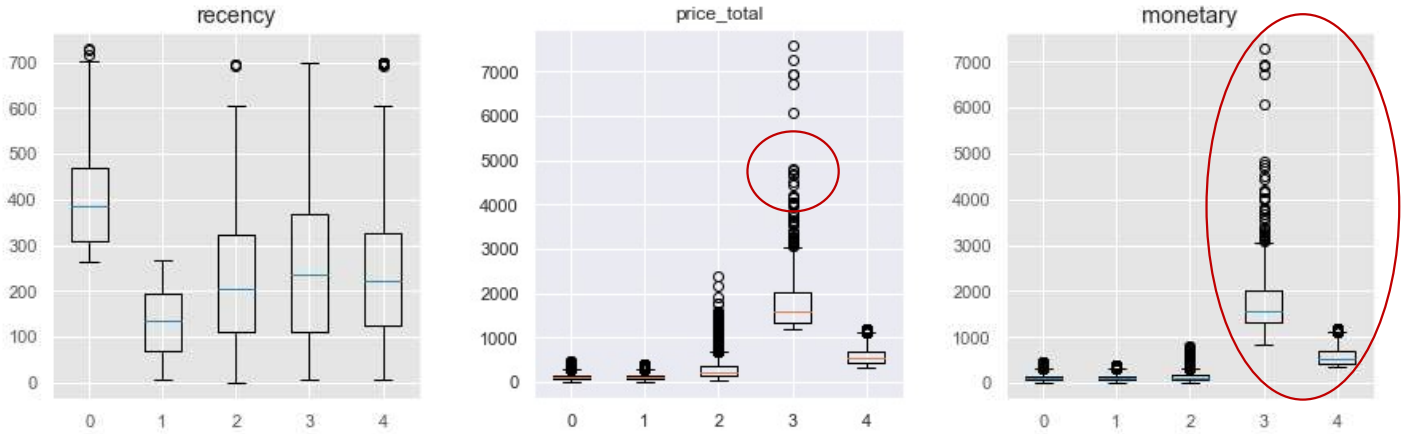
```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary




```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total



Nouvelles variables (2)

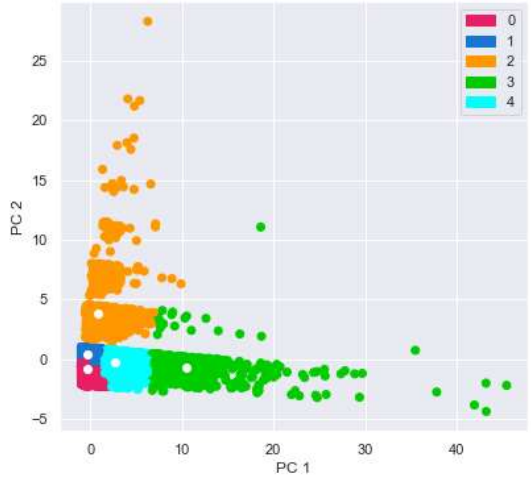
	recency	frequency	monetary	price_total	score
customer_unique_id					
0000366f3b9a7992bf8c76cfdf3221e2	116.0	1.0	141.9	141.9	5.0

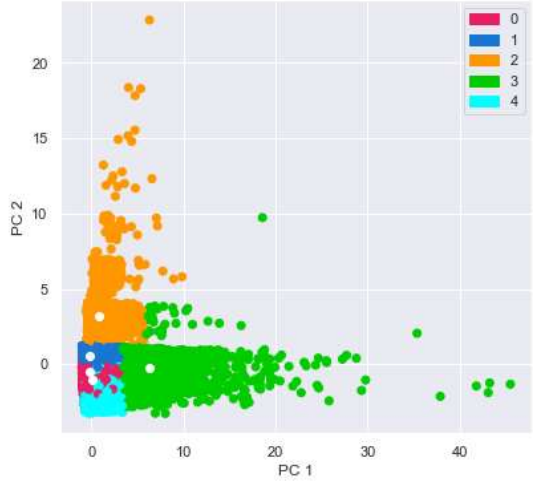
Visualisation des groupes (K-Means 5)

KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	396.12	1.00	117.99	36144
1	#1976D2	bleu	134.29	1.00	117.31	49721
2	#FF9800	orange	226.05	2.11	139.44	2893
3	#00CC00	vert	250.41	1.03	1784.87	788
4	#00FFFF	bleu (aqua)	236.55	1.00	586.79	5872

KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	398.27	1.00	130.70	31518
1	#1976D2	bleu	127.20	1.00	130.66	41798
2	#FF9800	orange	226.16	2.11	136.73	2880
3	#00CC00	vert	248.65	1.01	1141.03	2529
4	#00FFFF	bleu (aqua)	244.70	1.00	149.34	16693

```
data_c_plot(kmea5_l,kmea6_l,kmea5,kmea6)
```





recency	frequency	monetary	price_total

Nouvelles variables (2)

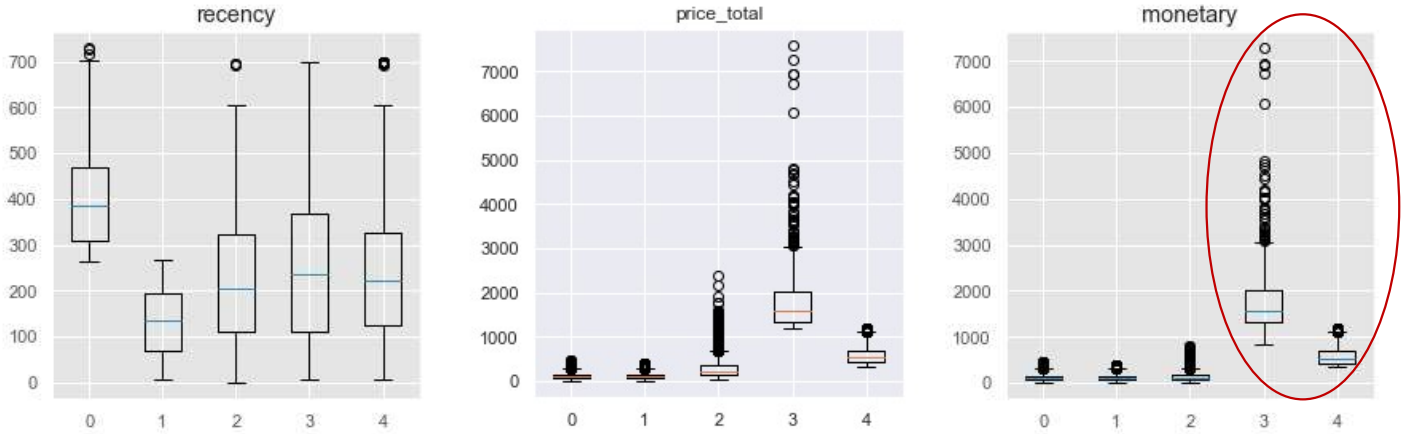
Statistiques

- K-Means 5 (4 var)

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total

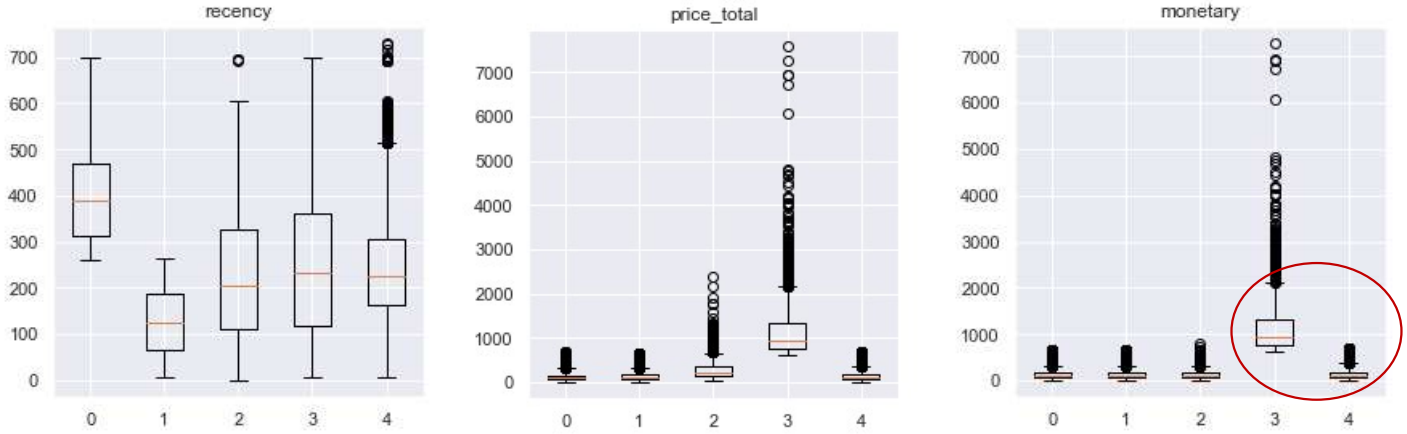


- K-Means 5 (5 var)

KM_group_5	Nombre de clients
0	31518
1	41798
2	2880
3	2529
4	16693

```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total score



Nouvelles variables (2)

Statistiques

- K-Means 5 (4 var)

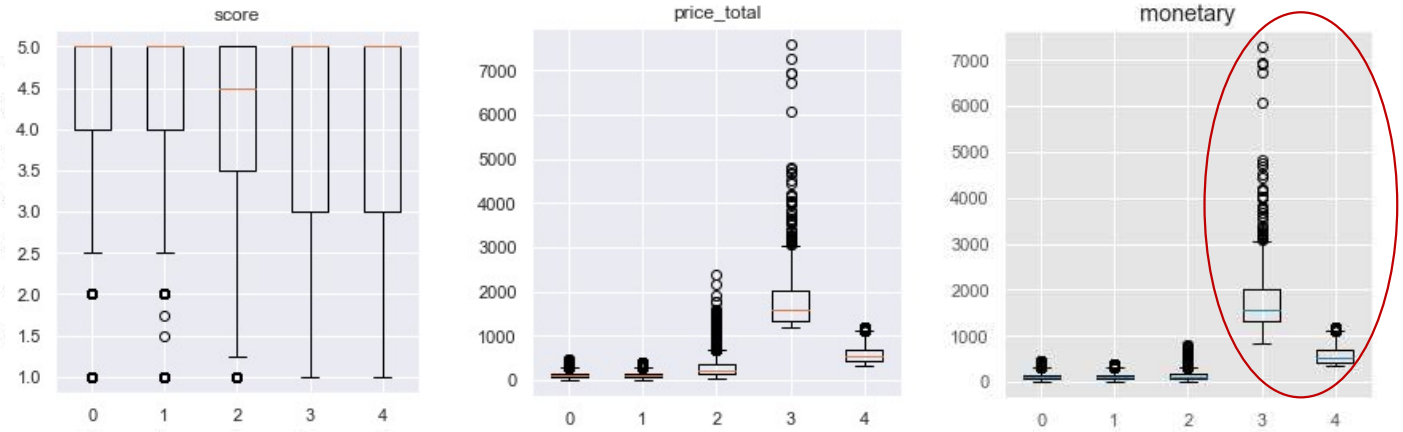
KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

- K-Means 5 (5 var)

KM_group_5	Nombre de clients
0	31518
1	41798
2	2880
3	2529
4	16693

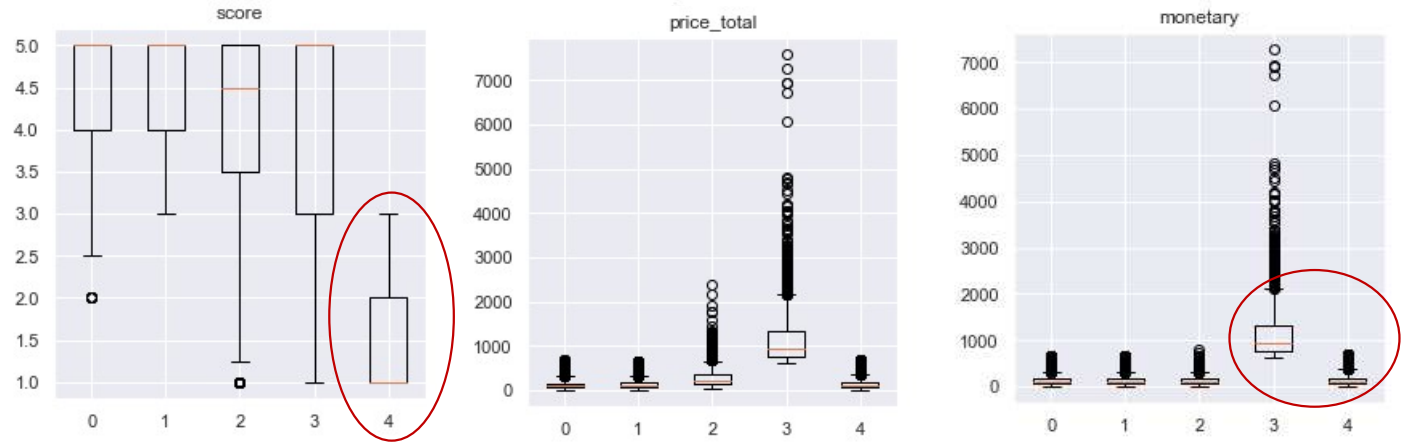
```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total




```
data_boxplot(df_rfm,'KM_group_5',1)
```

recency frequency monetary price_total score

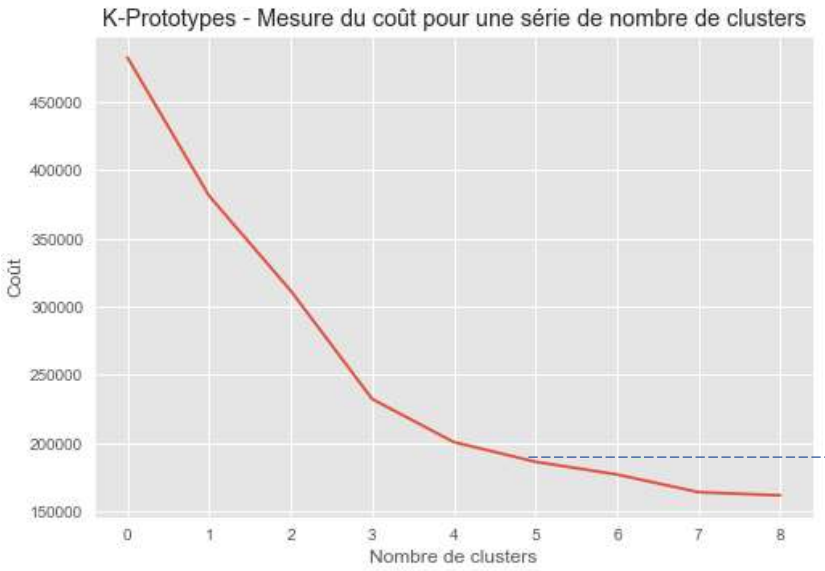


K-Prototypes - Données numériques et catégorielles

Détermination du nombre optimal de clusters

- Mesure du coût

```
for num_clusters in list(range(1,10)):

    kpro = KPrototypes(n_clusters = num_clusters,
                       init = 'Cao',
                       n_init = 1,
                       verbose = 0,
                       random_state = 42)

    kpro.fit_predict(X_c, categorical=[3,76])
```




```
Hyper-paramètres

{'cat_dissim': <function matching_dissim at 0x000002524796A730>,
 'gamma': 0.1277866255022374,
 'init': 'Cao',
 'max_iter': 100,
 'n_clusters': 5,
 'n_init': 1,
 'n_jobs': 1,
 'num_dissim': <function euclidean_dissim at 0x000002524796A7B8>,
 'random_state': 42,
 'verbose': 1}

Dernier coût = 200687.81920521837
Nombre d'itérations = 16
```

K-Prototypes

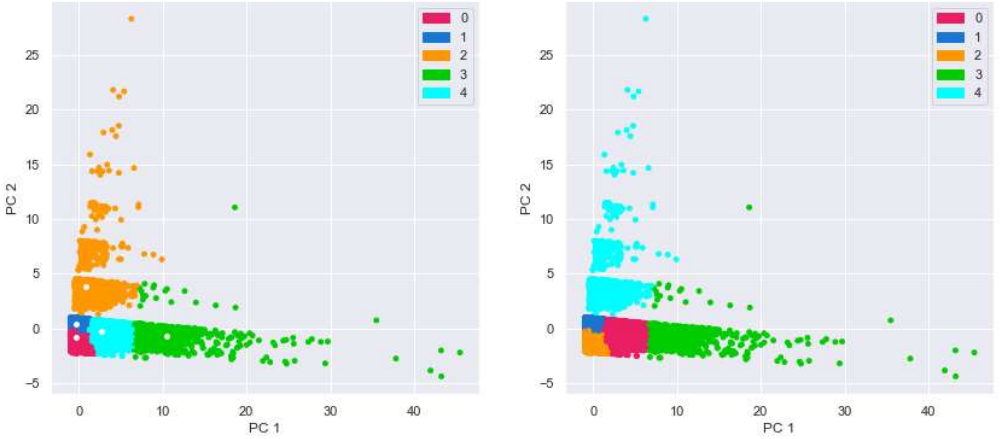
recency frequency monetary price_total

Visualisation des groupes (K-Means 5 vs K-Prototypes 5)

KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	396.12	1.00	117.99	36144
1	#1976D2	bleu	134.29	1.00	117.31	49721
2	#FF9800	orange	226.05	2.11	139.44	2893
3	#00CC00	vert	250.41	1.03	1784.87	788
4	#00FFFF	bleu (aqua)	236.55	1.00	586.79	5872

KPR_group	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	236.89	1.00	587.96	5846
1	#1976D2	bleu	133.87	1.00	117.38	49581
2	#FF9800	orange	395.51	1.00	118.07	36311
3	#00CC00	vert	250.38	1.03	1785.62	787
4	#00FFFF	bleu (aqua)	226.05	2.11	139.44	2893

```
data_c_plot(kmea5_l,kpro_l,kmea5)
```



K-Prototypes

Statistiques

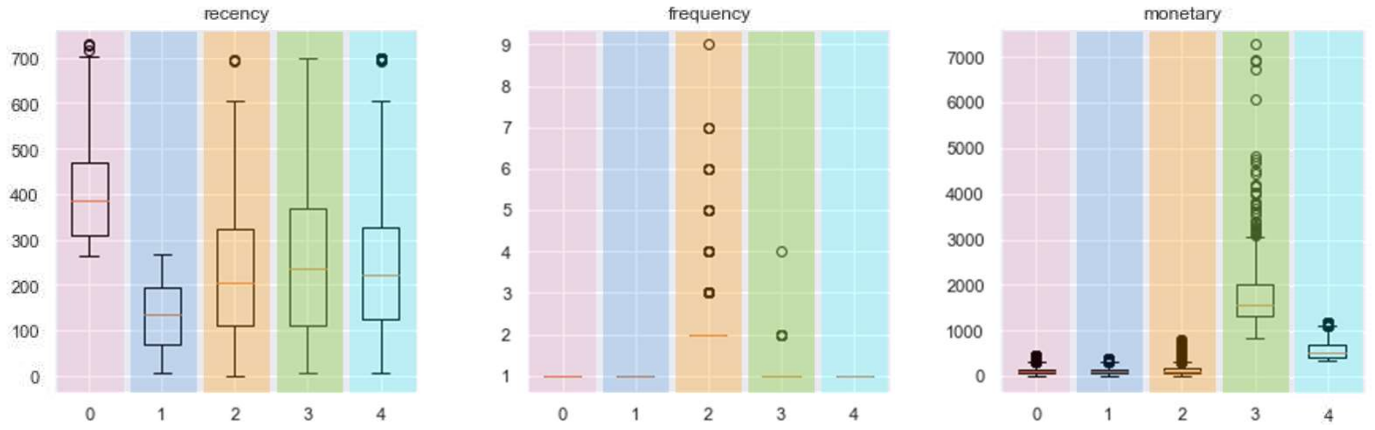
- K-Means 5

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

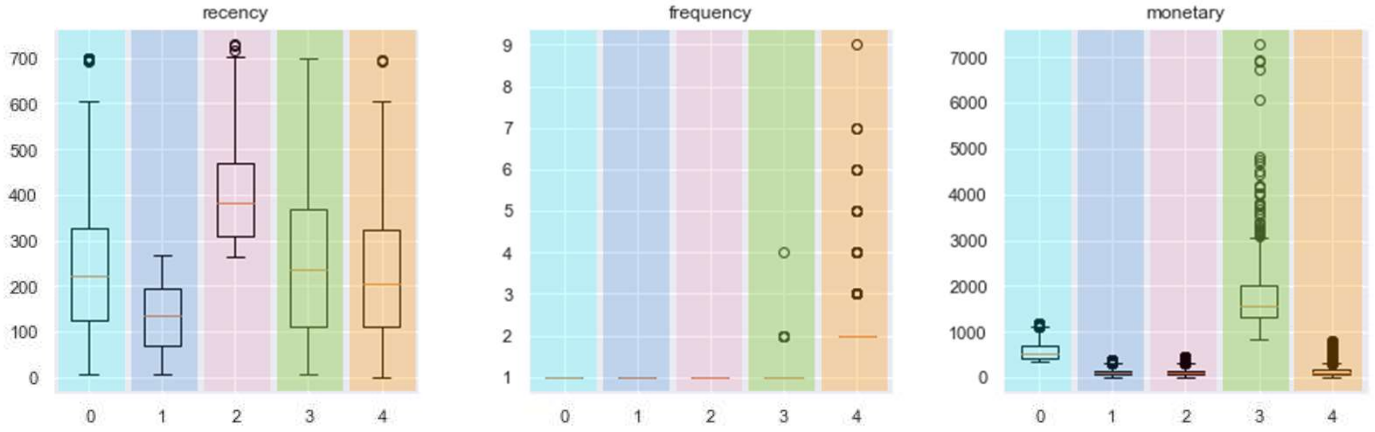
- K-Prototypes 5

KPR_group	Nombre de clients	
0	5846	- 26
1	49581	- 140
2	36311	+ 167
3	787	- 1
4	2893	

```
data_boxplot(df_rfm,'KM_group_5',1)
```




```
data_boxplot(df_rfm,'KPR_group',1)
```



K-Prototypes

Statistiques

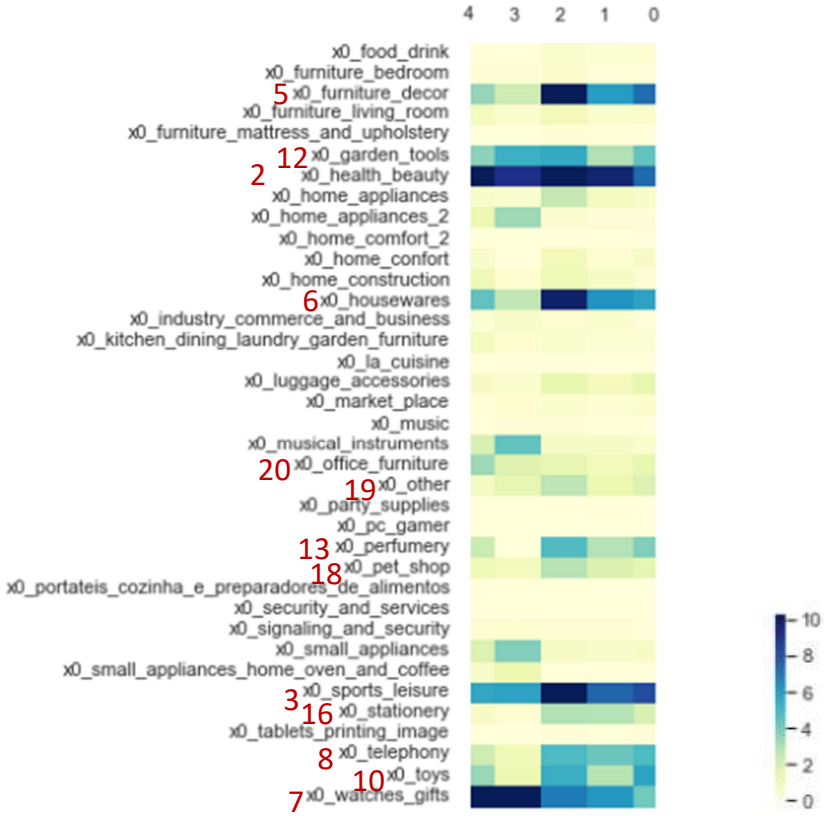
- K-Means 5

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

- K-Prototypes 5

KPR_group	Nombre de clients	
0	5846	- 26
1	49581	- 140
2	36311	+ 167
3	787	- 1
4	2893	

% de clients intéressés par la catégorie de produit



DBSCAN

Configuration

- Nombre de clusters souhaités: 5

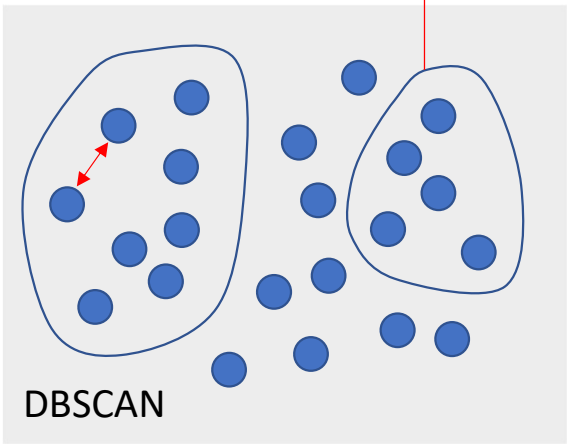
- Hyper-paramètres

```
dbs = DBSCAN(eps = 0.4, # Valeur avec X contenant 3 variables: 0.3
             min_samples = 25, # Valeur avec X contenant 3 variables: 20
             algorithm = 'auto').fit(X)
```

```
{'algorithm': 'auto',
 'eps': 0.4,
 'leaf_size': 30,
 'metric': 'euclidean',
 'metric_params': None,
 'min_samples': 25,
 'n_jobs': None,
 'p': None}
```

min_samples
nombre min de point dans un groupe

eps
distance entre 2 points



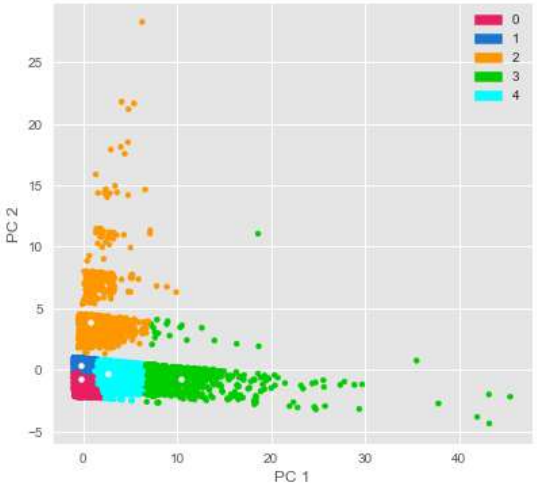
DBSCAN

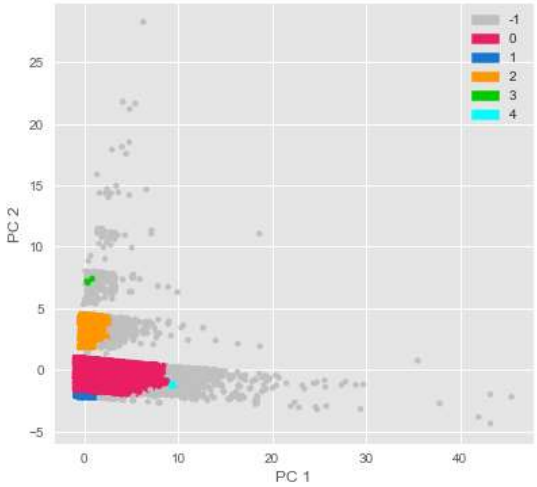
Visualisation des groupes (K-Means 5 vs DBSCAN)

KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	396.12	1.00	117.99	36144
1	#1976D2	bleu	134.29	1.00	117.31	49721
2	#FF9800	orange	226.05	2.11	139.44	2893
3	#00CC00	vert	250.41	1.03	1784.87	788
4	#00FFFF	bleu (aqua)	236.55	1.00	586.79	5872

DBS_group	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
-1	#C0C0C0	gris	270.87	1.79	1264.75	851
0	#E91E63	fuchsia	242.44	1.00	151.50	91746
1	#1976D2	bleu	696.80	1.00	125.38	263
2	#FF9800	orange	225.58	2.00	119.06	2500
3	#00CC00	vert	166.35	3.00	67.78	31
4	#00FFFF	bleu (aqua)	336.93	1.00	1640.56	27

```
data_c_plot(kmea5_l,dbs_l,kmea5)
```





Groupe 0: grande majorité de clients moyennement récents aux commandes ponctuelles à montants moyens

Groupe 3: petit groupe de clients très récents qui achètent souvent pour des petits montants

Groupe 4: petit groupe de clients moyennement récents aux commandes ponctuelles à grands montants

DBSCAN

Statistiques

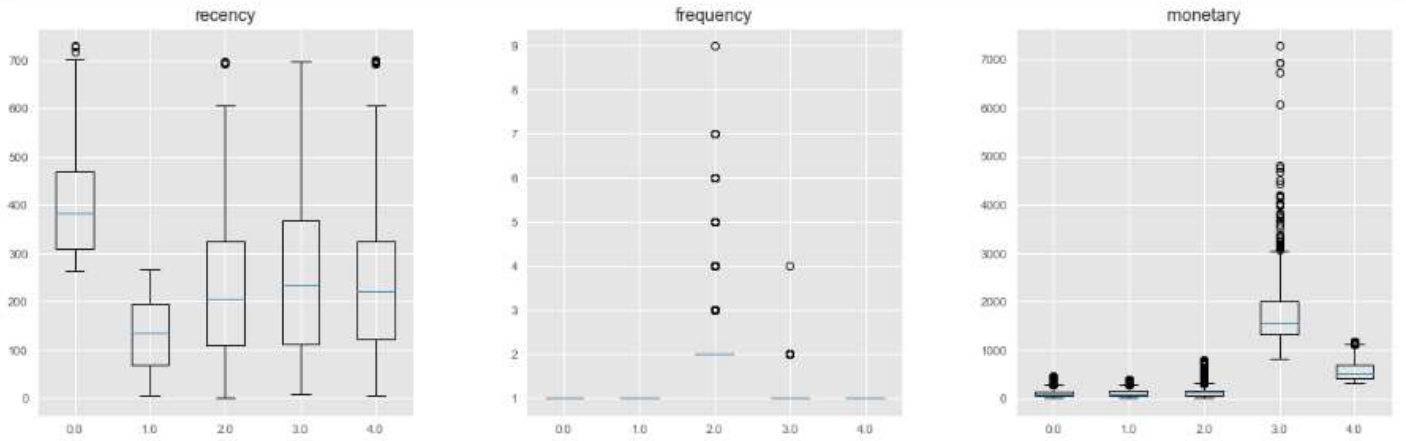
- K-Means 5

KM_group_5	Nombre de clients
0	36144
1	49721
2	2893
3	788
4	5872

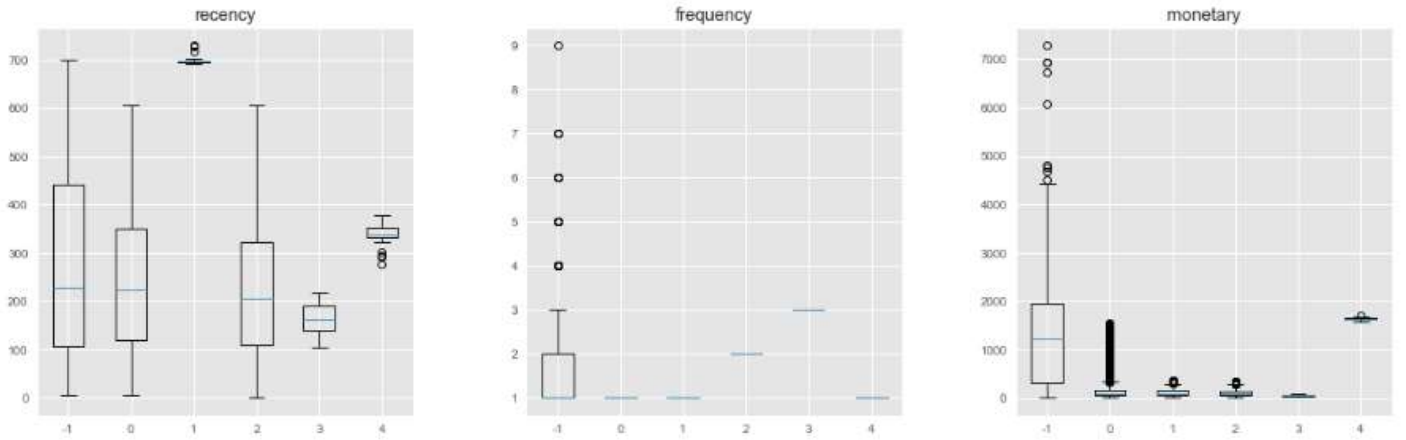
- DBSCAN

DBS_group	Nombre de clients
-1	851
0	91746
1	263
2	2500
3	31
4	27

```
data_boxplot(df_rfm,'KM_group_5',1)
```

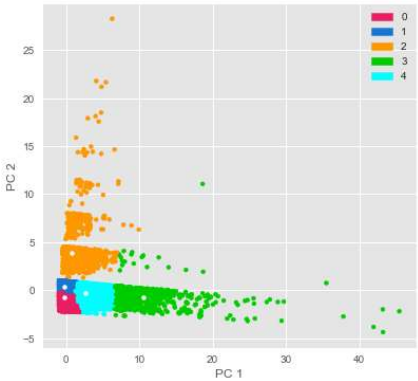



```
data_boxplot(df_rfm,'DBS_group',1)
```

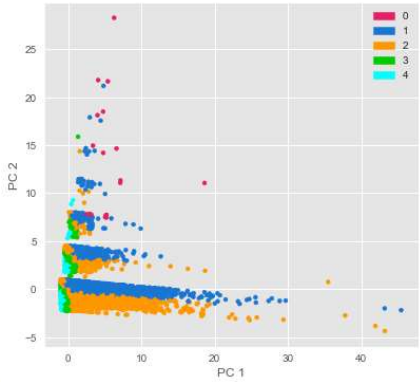


Comparaisons Clusters / Segments

K-Means 5



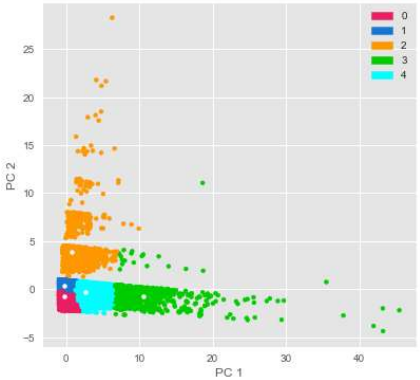
Segmentation RFM

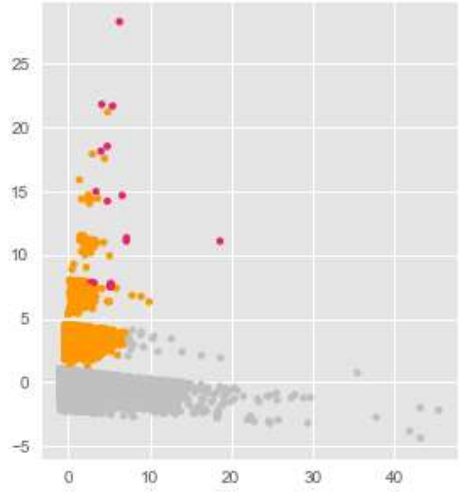


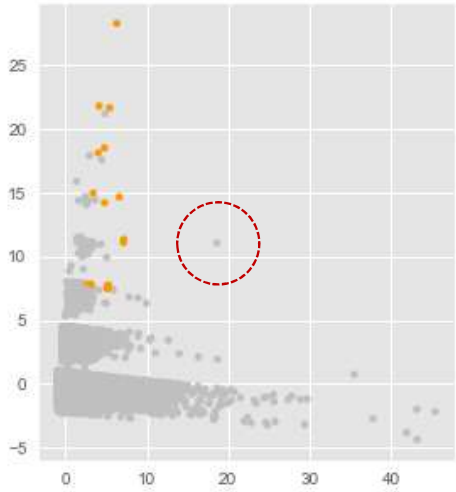
[0] Ancien - Ponctuel - Petit CA (36144) / [1] Très récent - Occasionnel - Grand CA (23183)	226
[0] Ancien - Ponctuel - Petit CA (36144) / [2] Récent - Occasionnel - Moyen CA (34199)	9294
[0] Ancien - Ponctuel - Petit CA (36144) / [3] Ancien - Occasionnel - Moyen CA (26455)	15314
[0] Ancien - Ponctuel - Petit CA (36144) / [4] Ancien - Occasionnel - Petit CA (11560)	11310
[1] Très récent - Ponctuel - Petit CA (49721) / [1] Très récent - Occasionnel - Grand CA (23183)	18086
[1] Très récent - Ponctuel - Petit CA (49721) / [2] Récent - Occasionnel - Moyen CA (34199)	21229
[1] Très récent - Ponctuel - Petit CA (49721) / [3] Ancien - Occasionnel - Moyen CA (26455)	10406
[2] Récent - Récurrent - Moyen CA (2893) / [0] Très récent - Récurrent - Grand CA (21)	20
[2] Récent - Récurrent - Moyen CA (2893) / [1] Très récent - Occasionnel - Grand CA (23183)	808
[2] Récent - Récurrent - Moyen CA (2893) / [2] Récent - Occasionnel - Moyen CA (34199)	1080
[2] Récent - Récurrent - Moyen CA (2893) / [3] Ancien - Occasionnel - Moyen CA (26455)	735
[2] Récent - Récurrent - Moyen CA (2893) / [4] Ancien - Occasionnel - Petit CA (11560)	250
[3] Récent - Occasionnel - Très grand CA (788) / [0] Très récent - Récurrent - Grand CA (21)	1
[3] Récent - Occasionnel - Très grand CA (788) / [1] Très récent - Occasionnel - Grand CA (23183)	436
[3] Récent - Occasionnel - Très grand CA (788) / [2] Récent - Occasionnel - Moyen CA (34199)	351
[4] Récent - Ponctuel - Grand CA (5872) / [1] Très récent - Occasionnel - Grand CA (23183)	3627
[4] Récent - Ponctuel - Grand CA (5872) / [2] Récent - Occasionnel - Moyen CA (34199)	2245

Comparaisons Clusters / Segments

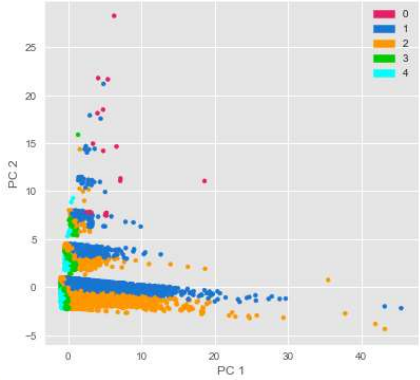
K-Means 5





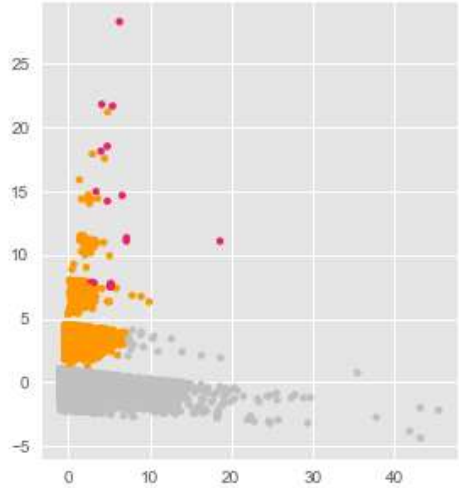


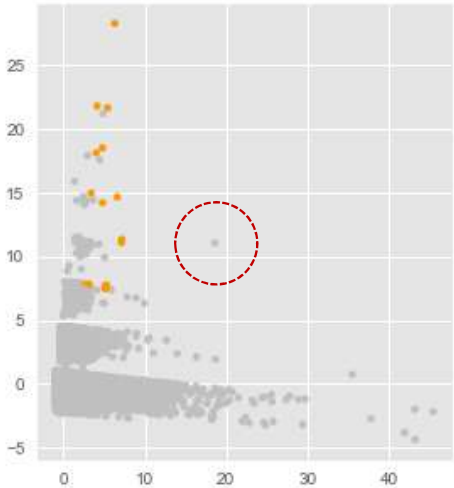
Segmentation RFM

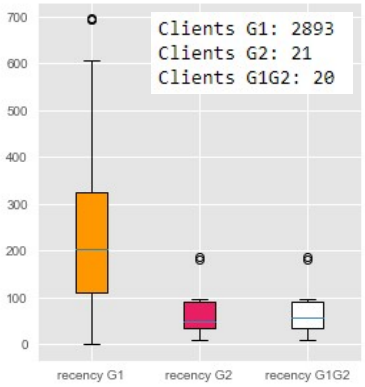


Comparaisons Clusters / Segments

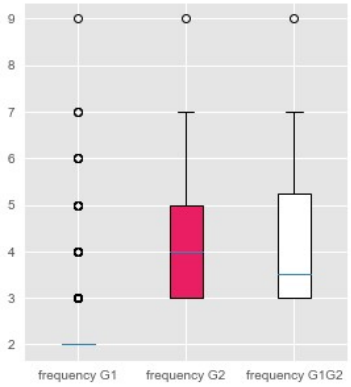
	customer_unique_id	recency	frequency (number of orders)	monetary
19108	33176de67c05eeed870fd49f234387a0	91.0	3.0	418.003333
21431	394ac4de8f3acb14253c177f0e15bc58	19.0	5.0	149.082000
21490	397b44d5bb99eabf54ea9c2b41ebb905	78.0	4.0	439.132500
23301	3e43e6105506432c953e165fb2acf44c	188.0	9.0	130.296667
23451	3ea4185126195c34ed9949d8e406d570	47.0	3.0	268.876667
29189	4e1cce07cd5937c69dacac3c8b13d965	35.0	3.0	431.533333
29773	4facc2e6fbc2bffab2fea92d2b4aa7e4	21.0	4.0	440.187500
35373	5e8f38a9a1c023f3db718edcf926a2db	181.0	5.0	222.836000
37319	63cfc61cee11cbe306bff5857d00bfe4	98.0	6.0	137.720000
37531	6469f99c1f9dfae7733b25662e7f1782	67.0	7.0	108.404286
44905	7849336c1b7785f258ffef06aedbd216	41.0	3.0	207.520000
51223	8961b4ca2c5aceb7a78ea72c6e0c840a	72.0	3.0	260.133333
66812	b2e9a05d23ea17713b5d7799f2004f8e	36.0	3.0	260.880000
69322	b9badb100ff8ecc16a403111209e3a06	29.0	3.0	246.943333
74750	c8460e4251689ba205045f3ea17884a1	26.0	4.0	1163.970000
75558	ca77025e7201e3b30c44b472ff346268	94.0	7.0	160.388571
78089	d132b863416f85f2abb1a988ca05dd12	44.0	3.0	425.506667
79004	d387ea85dc301a91740e31360d355686	98.0	3.0	430.293333
79447	d4a5e9f19897de65433c9d97bf4b9f8e	48.0	3.0	212.300000
82303	dc813062e0fc23409cd255f7f53c7074	11.0	6.0	172.270000
94861	fe81bb32c243a86b2f86fbf053fe6140	74.0	5.0	318.152000



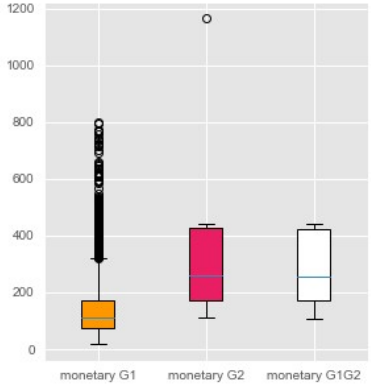




K-M 5 RFM



K-M 5 RFM



K-M 5 RFM

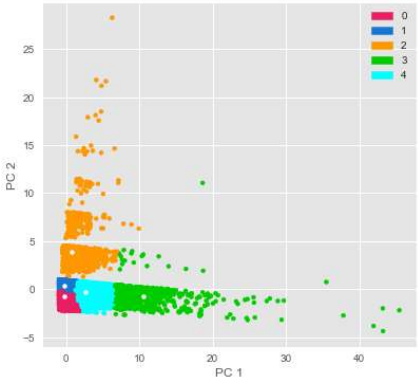
Comparaisons Clusters / Segments

	customer_unique_id	recency	frequency (number of orders)	monetary
19108	33176de67c05eeed870fd49f234387a0	91.0	3.0	418.003333
21431	394ac4de8f3acb14253c177f0e15bc58	19.0	5.0	149.082000
21490	397b44d5bb99eabf54ea9c2b41ebb905	78.0	4.0	439.132500
23301	3e43e6105506432c953e165fb2acf44c	188.0	9.0	130.296667
23451	3ea4185126195c34ed9949d8e406d570	47.0	3.0	268.876667
29189	4e1cce07cd5937c69dacac3c8b13d965	35.0	3.0	431.533333
29773	4facc2e6fbc2bffab2fea92d2b4aa7e4	21.0	4.0	440.187500
35373	5e8f38a9a1c023f3db718edcf926a2db	181.0	5.0	222.836000
37319	63cfc61cee11cbe306bff5857d00bfe4	98.0	6.0	137.720000
37531	6469f99c1f9dfae7733b25662e7f1782	67.0	7.0	108.404286
44905	7849336c1b7785f258ffef06aedbd216	41.0	3.0	207.520000
51223	8961b4ca2c5aceb7a78ea72c6e0c840a	72.0	3.0	260.133333
66812	b2e9a05d23ea17713b5d7799f2004f8e	36.0	3.0	260.880000
69322	b9badb100ff8ecc16a403111209e3a06	29.0	3.0	246.943333
74750	c8460e4251689ba205045f3ea17884a1	26.0	4.0	1163.970000
75558	ca77025e7201e3b30c44b472ff346268	94.0	7.0	160.388571
78089	d132b863416f85f2abb1a988ca05dd12	44.0	3.0	425.506667
79004	d387ea85dc301a91740e31360d355686	98.0	3.0	430.293333
79447	d4a5e9f19897de65433c9d97bf4b9f8e	48.0	3.0	212.300000
82303	dc813062e0fc23409cd255f7f53c7074	11.0	6.0	172.270000
94861	fe81bb32c243a86b2f86fbf053fe6140	74.0	5.0	318.152000

customer_unique_id	c8460e4251689ba205045f3ea17884a1
start	2018-08-07 00:00:00
end	2018-08-08 00:00:00
older	27
recency	26
duration	2
frequency (number of orders)	4
frequency_2 (number of products)	24
product number average	6
number of different products	1
number of payment types	1
monetary	1163.97
average price	193.995
total price	4655.88
seller loc = delivery loc	0
mean score	4
R	5
F	2
M	5
RFM	525
rfm_score	12

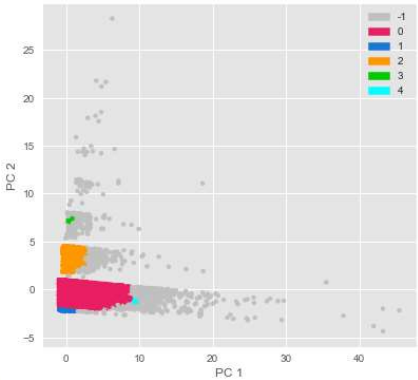
Comparaisons Clusters / Clusters

K-Means 5



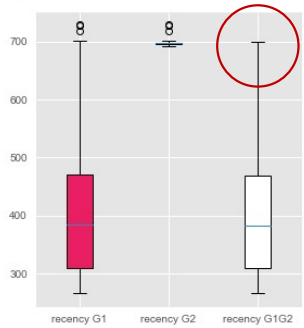
[0] Ancien - Ponctuel - Petit CA (36144) / [-1] Récent - Occasionnel - Grand CA (851)	11
[0] Ancien - Ponctuel - Petit CA (36144) / [0] Récent - Ponctuel - Moyen CA (91746)	35870
[0] Ancien - Ponctuel - Petit CA (36144) / [1] Inactif - Ponctuel - Moyen CA (263)	263
[1] Très récent - Ponctuel - Petit CA (49721) / [0] Récent - Ponctuel - Moyen CA (91746)	49721
[2] Récent - Récurrent - Moyen CA (2893) / [-1] Récent - Occasionnel - Grand CA (851)	362
[2] Récent - Récurrent - Moyen CA (2893) / [2] Récent - Récurrent - Moyen CA (2500)	2500
[2] Récent - Récurrent - Moyen CA (2893) / [3] Très récent - Récurrent - Petit CA (31)	31
[3] Récent - Occasionnel - Très grand CA (788) / [-1] Récent - Occasionnel - Grand CA (851)	444
[3] Récent - Occasionnel - Très grand CA (788) / [0] Récent - Ponctuel - Moyen CA (91746)	317
[3] Récent - Occasionnel - Très grand CA (788) / [4] Ancien - Ponctuel - Très grand CA (27)	27
[4] Récent - Ponctuel - Grand CA (5872) / [-1] Récent - Occasionnel - Grand CA (851)	34
[4] Récent - Ponctuel - Grand CA (5872) / [0] Récent - Ponctuel - Moyen CA (91746)	5838

DBSCAN

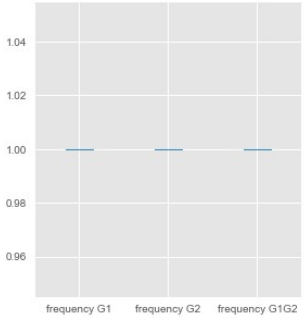



```
Clients G1: 36144
Clients G2: 263
Clients G1G2: 35881
```

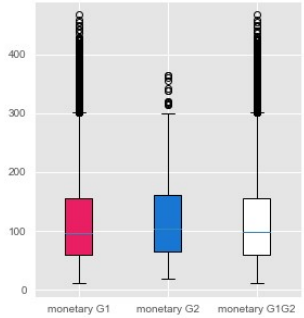
Suppression des inactifs



K-M 5 DBSCAN



K-M 5 DBSCAN

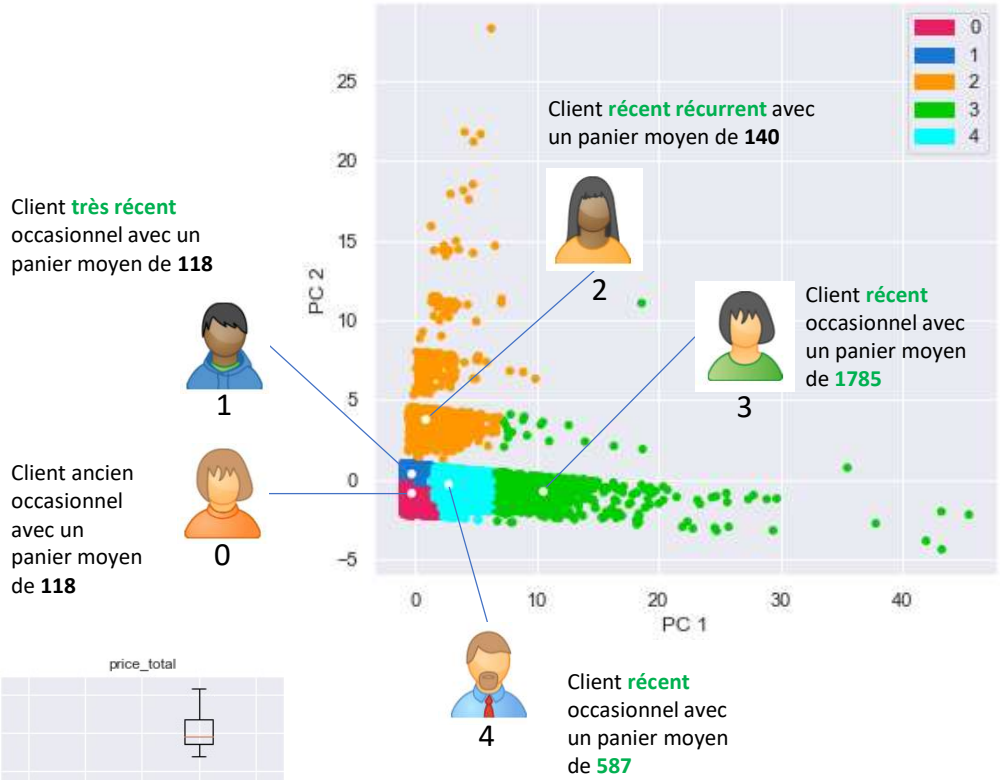


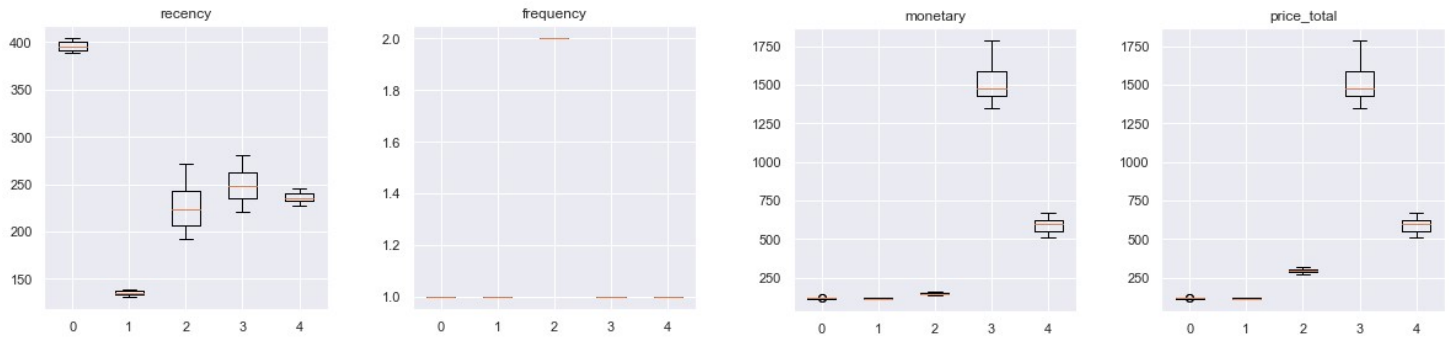
K-M 5 DBSCAN

Personas

- K-Means 5 clusters
- 4 Variables

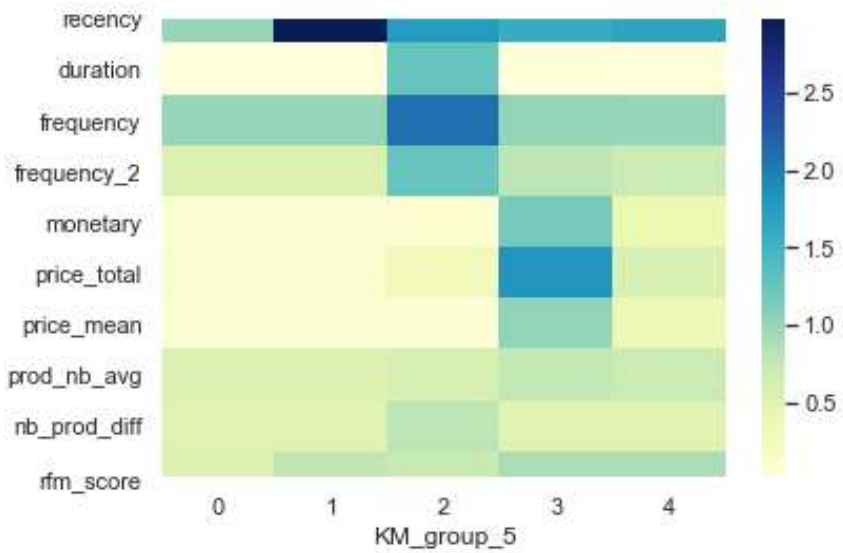
KM_group_5	code couleur	couleur	recency_mean	frequency_mean	monetary_mean	Nombre de clients
0	#E91E63	fuchsia	396.12	1.00	117.99	36144
1	#1976D2	bleu	134.29	1.00	117.31	49721
2	#FF9800	orange	226.05	2.11	139.44	2893
3	#00CC00	vert	250.41	1.03	1784.87	788
4	#00FFFF	bleu (aqua)	236.55	1.00	586.79	5872

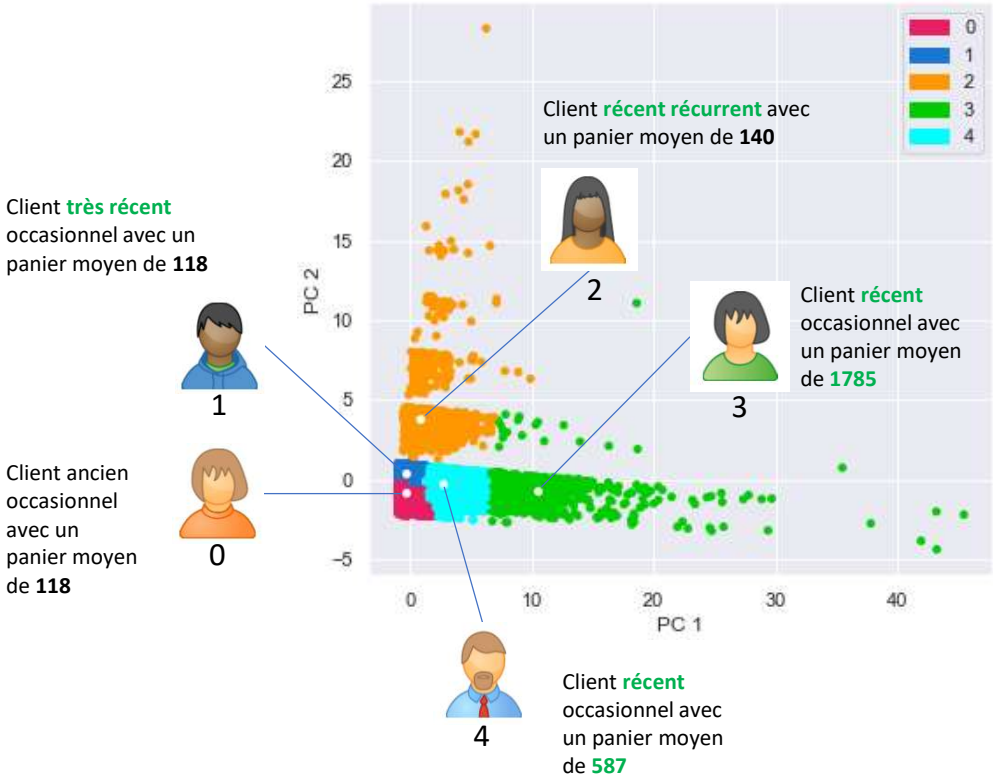




Personas

- K-Means 5 clusters



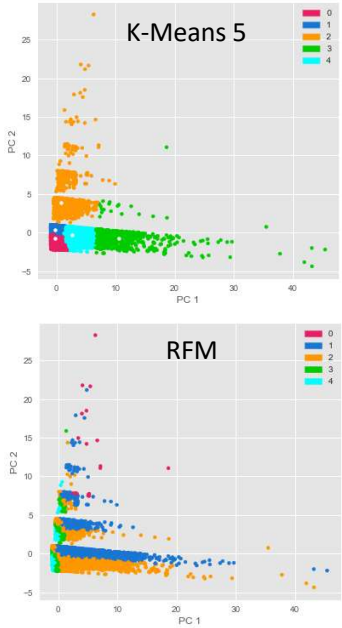


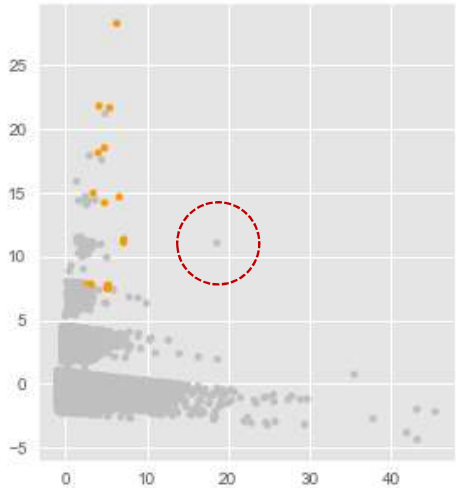
Géolocalisation

Validation des coordonnées géographiques

```
Nombre de clients sans code de geolocalisation: 264
Nombre de clients total: 95417
Nombre de clients non géolocalisés: 95153
```

```
Clients non géolocalisés groupe 0: 0
```





Temps de traitements

Durée des opérations

	Opération	Durée	Temps réel
0	Initialisation	1.86	
1	Variables RFM	1.16	
2	Filtrage des individus (clients)	0.11	
3	Statistiques - RFM	2.61	
4	Segmentation - RFM	4.27	
5	Segmentation stratégique - RFM	1.68	
6	Jeu de données final - RFM	0.23	
7	Initialisation - Clustering	1.57	
8	Standardisation des données	0.06	
9	Analyse en Composantes Principales	0.28	
10	Paramètres graphiques	0.03	
11	Mesure distortions (1) - k-means	0.41	33.26
12	Mesure distortions (2) - k-means	21.65	
13	Variances groupes (calinski_harabasz) - k-means	22.33	
14	Coefficient de silhouette (1) - k-means	0.33	688.20
15	Coefficient de silhouette (2) - k-means	0.02	821.33 s
16	Stabilité (1) - k-means	0.07	92.92
17	Stabilité (2) - k-means	53.22	167.87
18	Clustering - k-means (3 cls)	0.31	1.47
19	Clustering - k-means (4 cls)	0.31	2.5
20	Clustering - k-means (5 cls)	0.29	2.67
21	Clustering - k-means (6 cls)	0.52	3.49
22	Clustering - k-means (7 cls)	0.48	3.67
23	Statistiques - k-means (5 cls)	31.82	
24	Groupes explicites - k-means (5 cls)	0.19	
25	Outliers - k-means (5 cls)	0.19	
26	Personas - k-means (5 cls)	27.22	85.27
27	Statistiques - RFM vs k-means	10.65	
28	Nombre de clusters - k-modes	0.39	651.38
29	Clustering - k-modes (5 cls)	0.15	71.37
30	Statistiques - k-modes (5 cls)	5.49	
31	Groupes explicites - k-modes (5 cls)	0.10	
32	Nombre de clusters - k-prototypes	0.41	1970.06
33	Clustering - k-protoypes (5 cls)	14.36	214.95
34	Statistiques - k-protoypes (5 cls)	14.78	
35	Groupes explicites - k-protoypes (5 cls)	0.07	
36	Clustering - DBSCAN	0.13	25.94
37	Statistiques - DBSCAN	18.34	
38	Groupes explicites - DBSCAN	0.08	
39	3D	40.22	
40	3D éclatée	21.80	
41	Geolocalisation	7.61	
42	Fonction recherche par client	7.83	
43	Partionnement final	8.93	

Segmentation RFM vs Clustering

Segmentation RFM

- Avantages
  - Facile à mette en œuvre techniquement (programme basique)
  - Contrôle de l'importance des variables (choix des variables, pondération)
  - Segmentation métier à façon (stratégie marketing)
- Inconvénients
  - Temps d'ajustement des paramètres
  - Fiabilité du Score

1) Utile pour comprendre le domaine

2) Sert de référence

3) Stratégie avant Segmentation

Clustering

- Avantages
  - Nombreuses possibilités techniques (librairies spécialisées)
  - Identification d'outliers et groupes marginaux
- Inconvénients
  - Pas de contrôle de l'importance des variables
  - Temps de traitements
  - Instabilité

1) Partionnement automatique

2) Groupes + homogènes qu'avec RFM

3) Stratégie après Clustering

8 Recommandations

Modèles de Clustering

K-Means

- Avantages
  - Durées de traitements rapides
- Inconvénients
  - Ne traite pas les données catégorielles

K-Prototypes

- Avantages
  - Prend en compte les données numériques et catégorielles
- Inconvénients
  - Durées de traitements longues

DBSCAN

- Avantages
  - Prend en compte les données numériques et catégorielles
- Inconvénients
  - Durées de traitements longues

Expérience utilisateur

Partitions homogènes aisées à exploiter

Pas d'apport particulier

Apporte des informations complémentaires

Base de clients

Mise à jour de la base de clients

- Nécessite un complément d'analyse et une revue des process

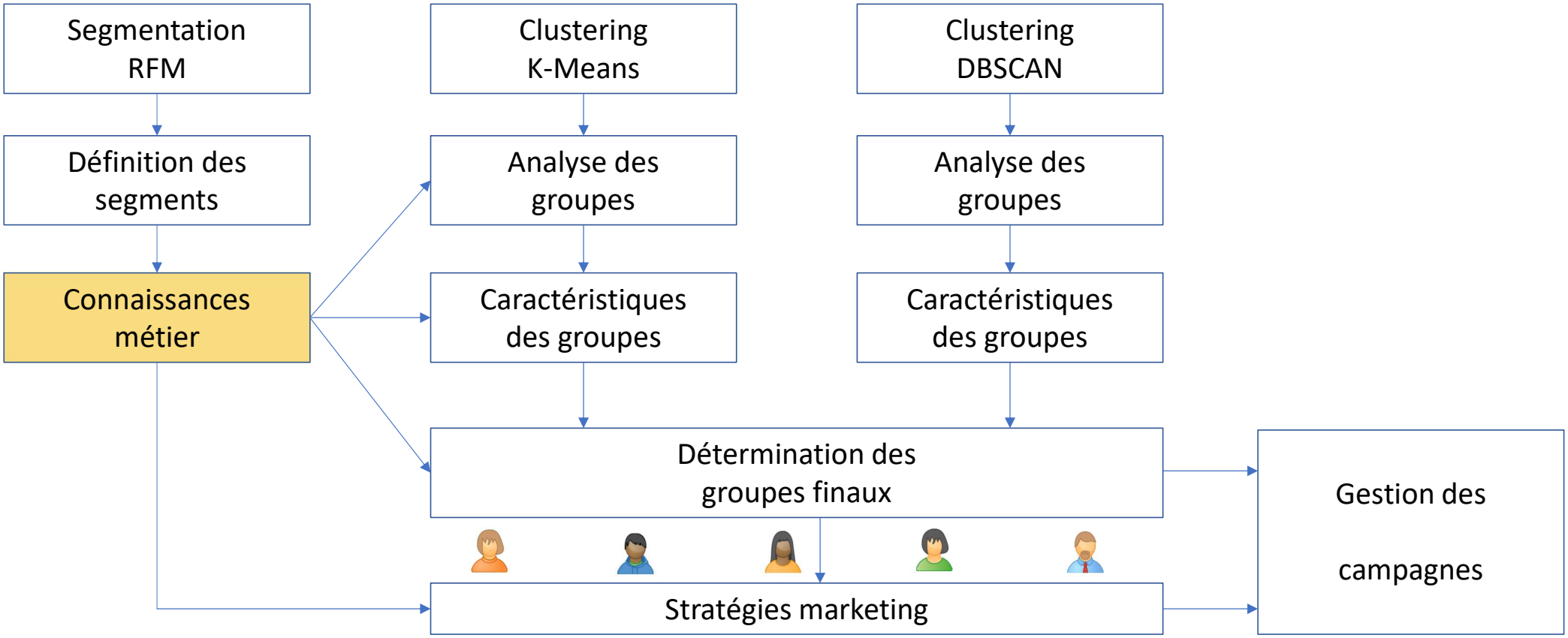
Modélisation des données

- Etapes de traitements automatisées

Clustering

- Analyse préliminaire
  - Distributions des variables
- Choix des variables
  - Pertinence et intérêt métier
- Evaluation de la stabilité
  - Nombre optimal de partitions
- Choix des hyper-paramètres
  - Nombre de clusters, n-init, random state…

Technologie au service du Métier



11 Prochaines étapes

Travail en équipe

Validation des premiers résultats

- Retenir ce qui est intéressant
- Relever ce qui manque

Révision des objectifs

- Orientations des stratégies marketing

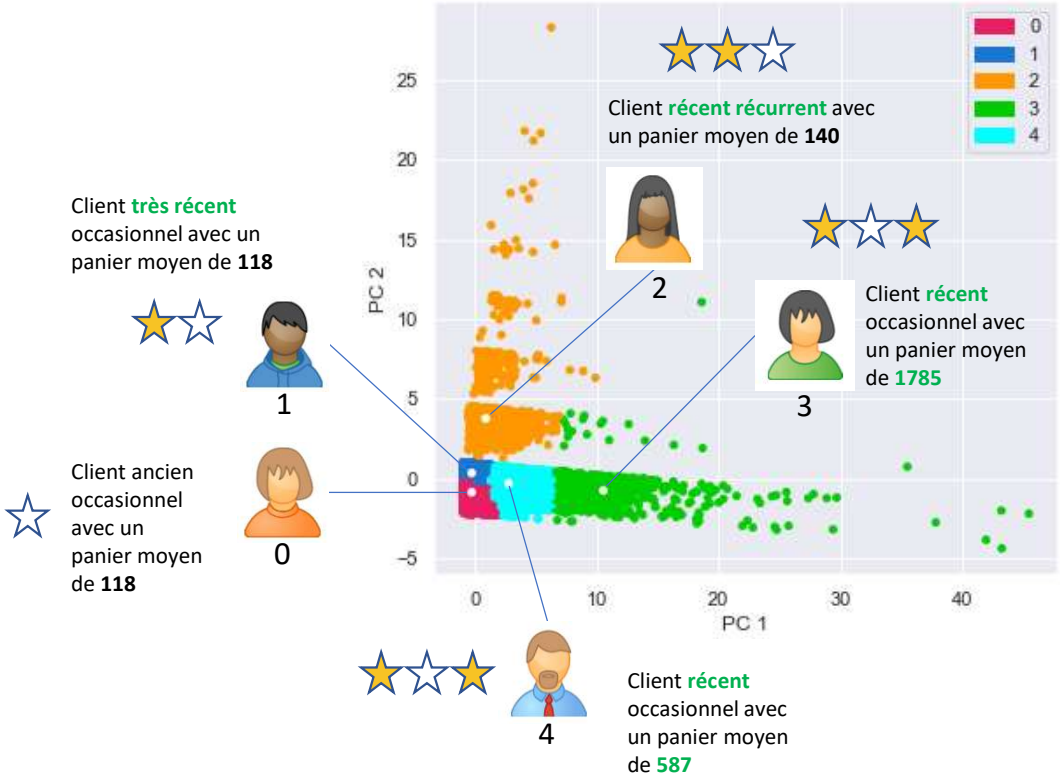
Analyse métier détaillé

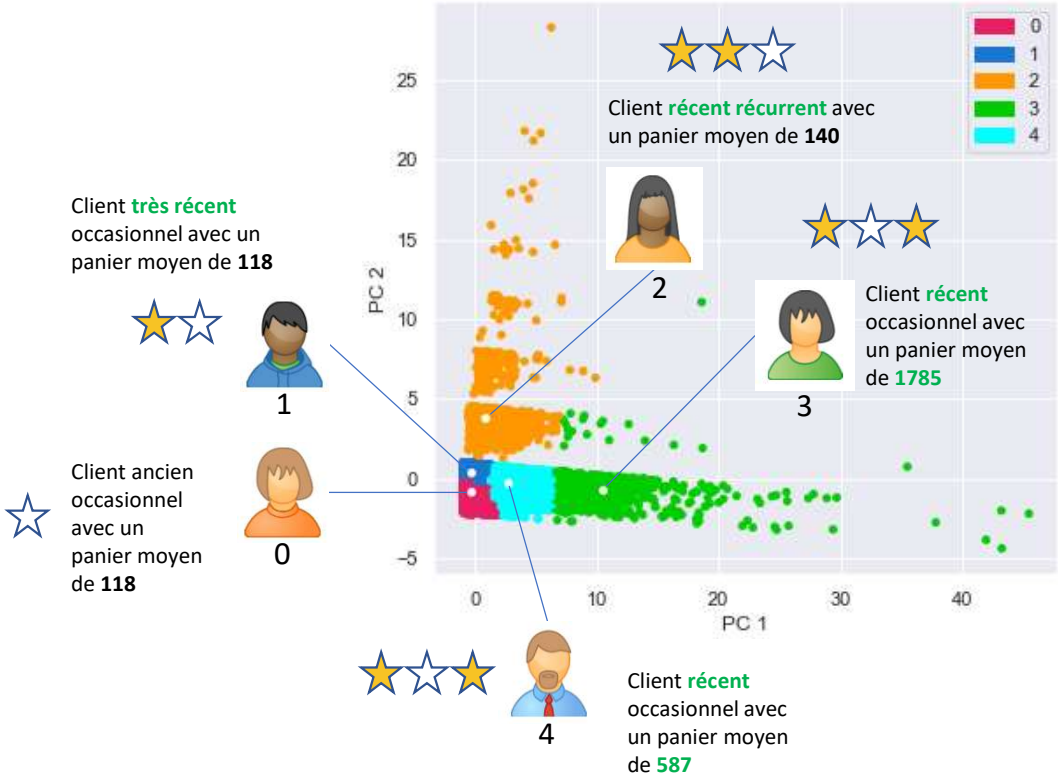
- Complément connaissances métier
- Choix des variables

Optimisation des algorithmes

- Choix des algorithmes
  - K-Means, DBSCAN, OPTICS…
- Ajustement des hyper-paramètres

★ Stratégies marketing



Merci!

★ Stratégies marketing