OPENCLASSROOMS

SEGMENTEZ DES CLIENTS D'UN SITE E-COMMERCE

P5. FARID DJADDA

20/04/2024

olist

PRÉSENTATION DE OLIST

OBJECTIFS DU PROJET

Objectif du Projet

L'objectif de ce projet est de développer une stratégie de segmentation des clients d'Olist pour renforcer l'efficacité des campagnes marketing, améliorer l'expérience utilisateur et optimiser la fidélisation de la clientèle.

Problématique

La problématique centrale à laquelle nous répondons est la suivante : **Comment pouvons-nous identifier et caractériser les différents segments de clients d'Olist pour adresser des communications ciblées et créer des offres personnalisées ?**

olist

Démarche et Méthodologie du Projet

1. Compréhension des Données
2. Analyse Exploratoire
3. Préparation des Données pour la Modélisation
4. Sélection et Entraînement des Modèles
5. Évaluation et Interprétation des Clusters
6. Analyse de la Stabilité des Clusters

olist

Collecte et Préparation des Données

Notre analyse s'appuie sur une combinaison de sources :

Sources de Données :

- Base de données relationnelle : Stocke l'historique complet des transactions et interactions clients.
- Fichiers CSV : Comprend des snapshots périodiques de données clients pour faciliter les analyses répétitives et le partage des informations.

Aperçu des Données :

	customer_unique_id	recence	frequence	montant_total	score_review_minimum
0	0000366f3b9a7992bf8c76cfdf3221e2	153.132905	1	129.90	5
1	0000b849f77a49e4a4ce2b2a4ca5be3f	159.248125	1	18.90	4
2	0000f46a3911fa3c0805444483337064	559.389734	1	69.00	3
3	0000f6ccb0745a6a4b88665a16c9f078	349.108970	1	25.99	4
4	0004aac84e0df4da2b147fca70cf8255	323.035463	1	180.00	5

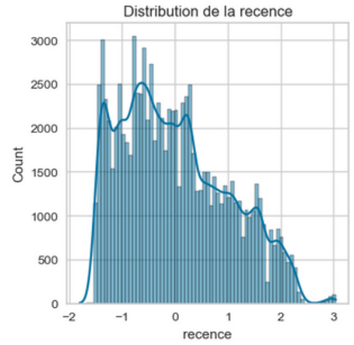
Préparation des Données :

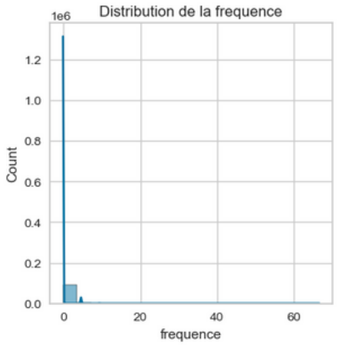
Le processus de préparation a inclus :

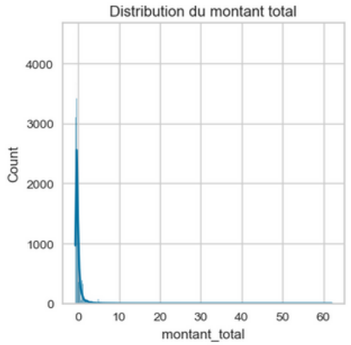
- Nettoyage des données
- Traitement des valeurs manquantes
- Normalisation
- Encodage des catégories

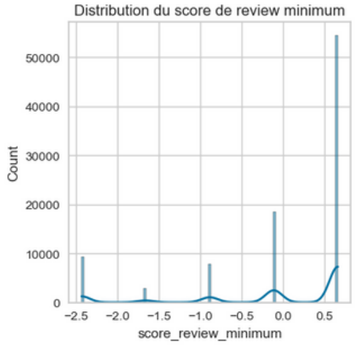
Exploration

Dans le cadre de mon exploration, J'ai choisi de présenter des graphiques illustrant la distribution et la relation entre les différentes caractéristiques des clients. Voici une analyse de ces graphiques:







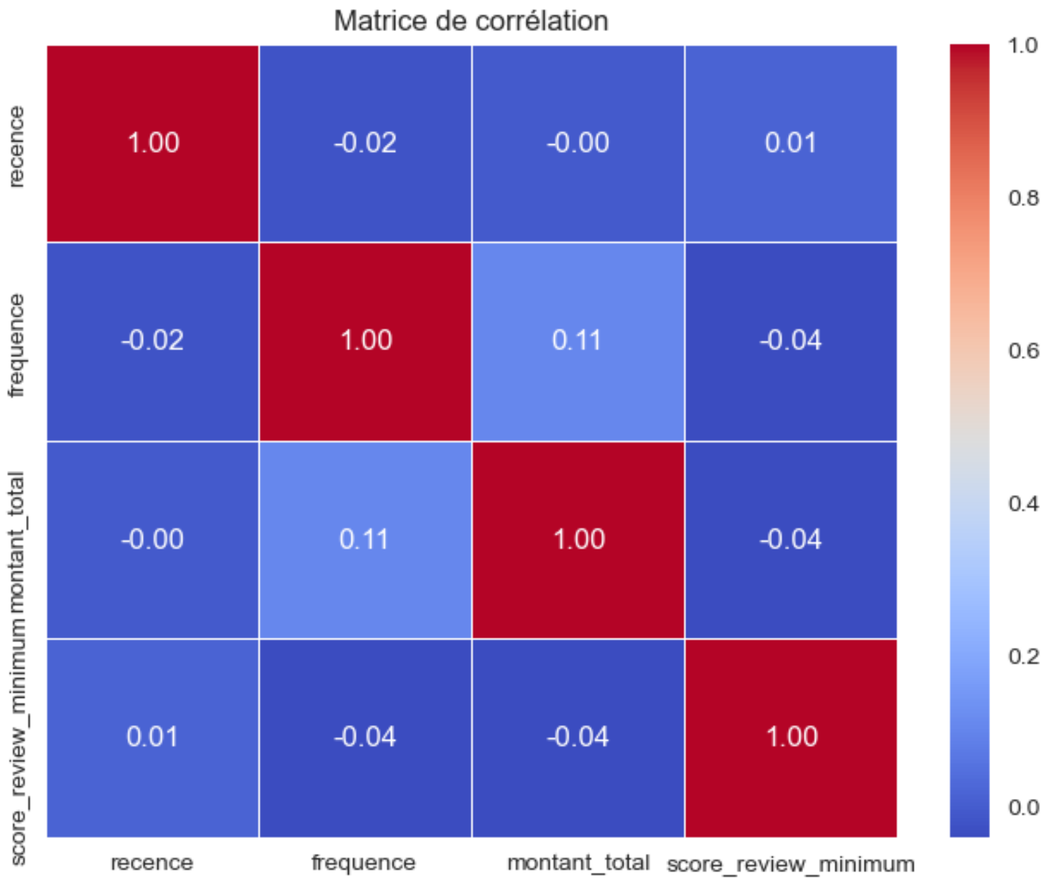


1. Distribution de la récence : représente le nombre de jours écoulés depuis le dernier achat d'un client.
2. Distribution de la fréquence : c'est-à-dire le nombre total d'achats qu'un client a effectués sur une période donnée.
3. Distribution du montant total : expose la distribution du montant total dépensé par les clients.
4. Distribution du score de review minimum : qui est un indicateur de la satisfaction client.

Exploration

<u>Matrice de corrélation :</u>

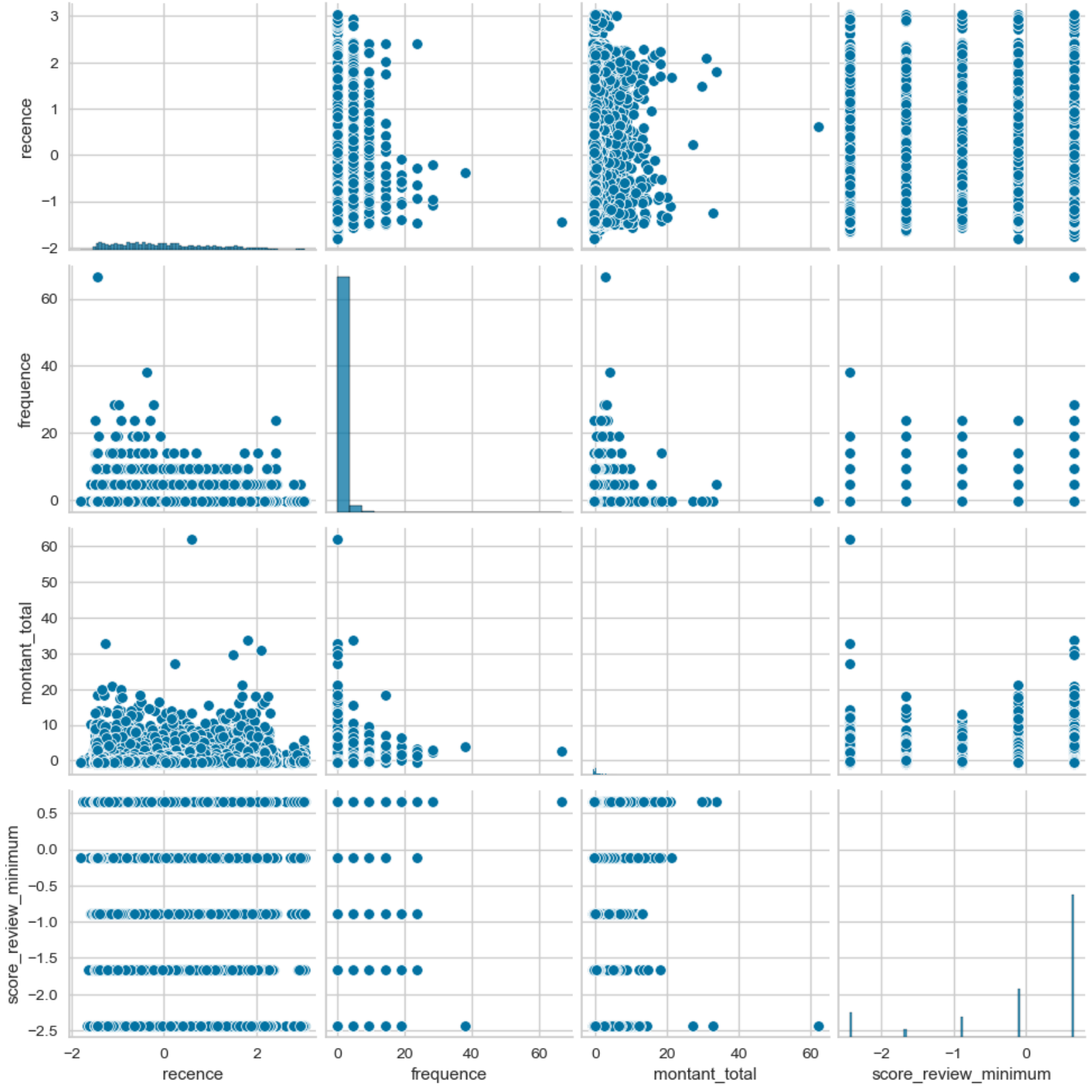
Le graphique de la matrice de corrélation nous permet d'examiner l'existence de relations entre les différentes caractéristiques des clients. Des corrélations fortes pourraient indiquer des tendances ou des comportements spécifiques au sein de notre base de clients.



Exploration

Graphiques de dispersion :

1. **Clusters et densité**
2. **Tendances**
3. **Répartition des données**



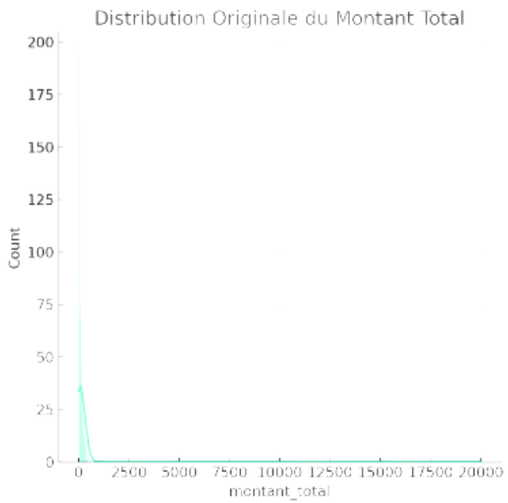
Prétraitement des Données

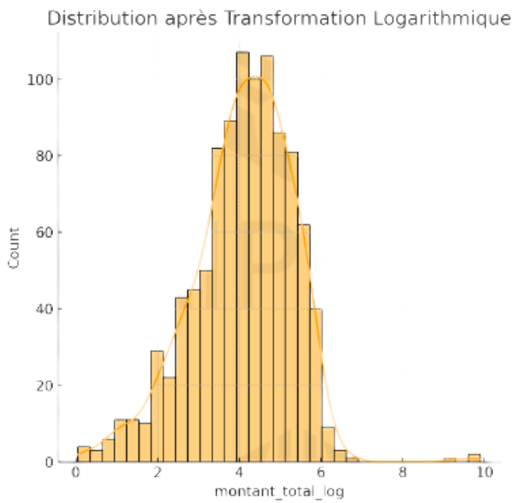
1

Transformations mathématiques appliquées :

Utilisation de techniques comme la transformation logarithmique pour réduire l'effet des valeurs extrêmes et améliorer la normalité des distributions.

Pour les variables sélectionnées, j'ai appliqué **np.log1p** (qui applique **log(1 + x)**, utile pour éviter les erreurs avec des valeurs de **0**).





Prétraitement des Données

2

Traitement des valeurs manquantes :

Les données ont été soigneusement examinées pour identifier et traiter les valeurs manquantes.

Dans notre jeu de données aucunes valeurs manquante à signaler.

```python
# Vérifier les valeurs manquantes
missing_values = customer_file.isnull().sum()
print(missing_values)
```

```
customer_unique_id      0
recence                 0
frequence               0
montant_total           0
score_review_minimum    0
dtype: int64
```

3

Normalisation :

Application de la normalisation StandardScaler pour que les variables numériques contribuent de manière équivalente aux algorithmes de modélisation.

Importance particulière dans le cadre de notre modélisation non supervisée, permettant de prévenir tout biais dû à l'échelle des variables.

	recence	frequence	montant_total	score_review_minimum
0	-0.789847	-0.160071	-0.053605	0.663454
1	-0.749598	-0.160071	-0.571158	-0.109947
2	1.884037	-0.160071	-0.337560	-0.883349
3	0.500020	-0.160071	-0.538100	-0.109947
4	0.328411	-0.160071	0.179993	0.663454

Modélisation

(Stratégie et Entraînement)

1. **Choix des algorithmes :**

   Les principaux algorithmes envisagés **incluaient K-Means**, **Clustering Hiérarchique et DBSCAN**, chacun ayant ses propres avantages en fonction des spécificités des données et des objectifs de l'analyse.

2. **Stratégie pour déterminer le nombre de clusters :**

   Méthode du coude et score de silhouette.

3. **Entraînement des modèles**

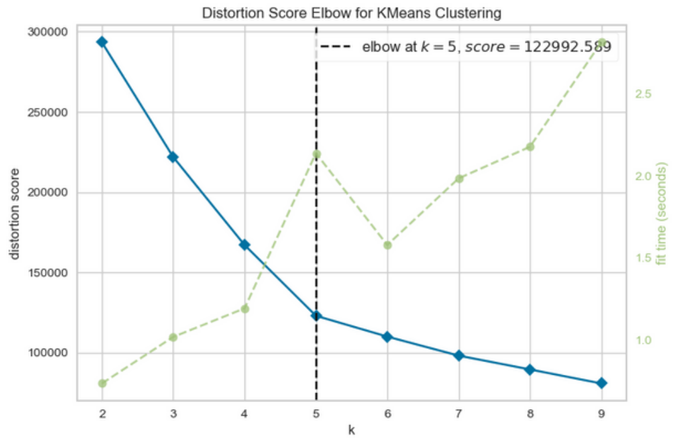
Stratégie pour déterminer le nombre de clusters

1

méthode du coude :

L'une des décisions les plus importantes dans le clustering par K-Means est le choix du nombre de clusters (k). J'ai utilisé la méthode du coude pour identifier le nombre optimal de clusters, ce qui nous permettra de trouver un équilibre entre la précision du modèle et sa complexité.

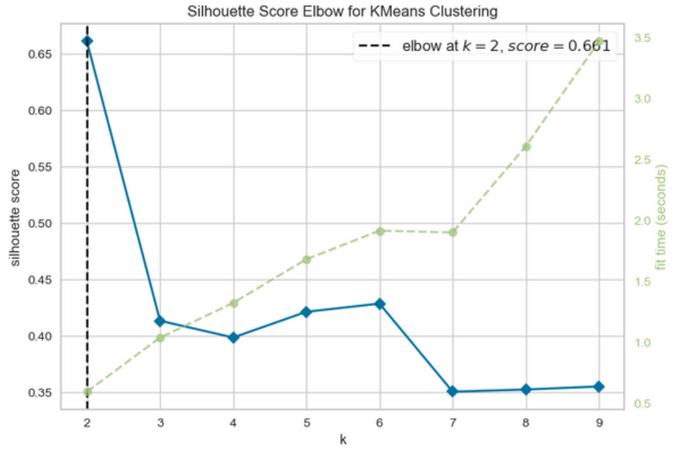
Dans notre projet, la méthode du coude a révélé un "coude" marqué lorsque **k = 5**



2

Score de silhouette :

Il faut considérer le contexte métier lors de la sélection du nombre de clusters. Même si le score de silhouette pour **k = 2** est le plus élevé, il pourrait être trop simpliste pour capturer la complexité et la variabilité des comportements des clients dans un contexte e-commerce.
Des clusters supplémentaires peuvent aider à distinguer des nuances plus fines dans les données qui pourraient être pertinentes pour Olist.



Entraînement des modèles

Les modèles ont été entraînés en normalisant d'abord les données pour assurer une égale pondération de chaque caractéristique. Nous avons procédé à plusieurs itérations d'entraînement, en affinant à chaque fois les paramètres et en réévaluant les performances grâce aux métriques de qualité. Le modèle final a été sélectionné sur la base de sa capacité à former des segments distincts et interprétables.

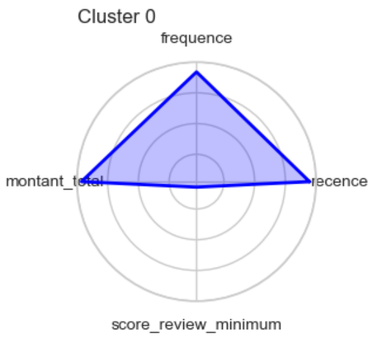
1. **K-Means**
2. **Clustering Hiérarchique**
3. **DBSCAN**

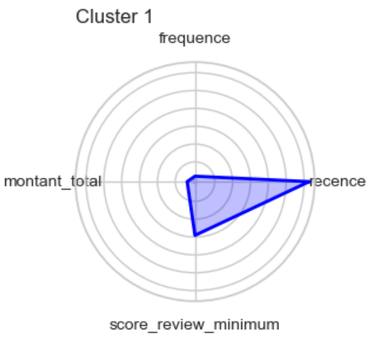
Entraînement des modèles

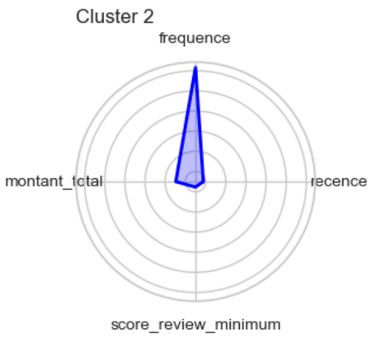
(Visualisation des clusters)

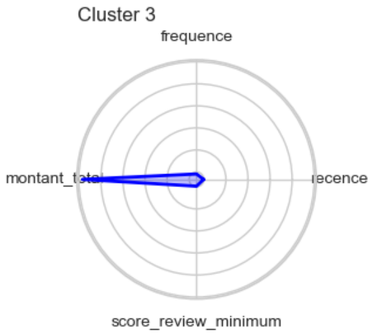
1 **K-Means :**

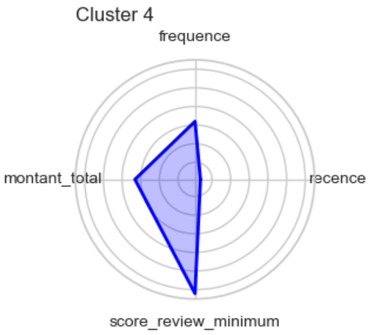
- Cluster 0 - "Les Nouveaux Prometteurs"
- Cluster 1 - "Les Clients Ponctuels"
- Cluster 2 - "Les Acheteurs Enthousiastes"
- Cluster 3 - "Les Clients Prestige"
- Cluster 4 - "Les Ambassadeurs Satisfaits"











Entraînement des modèles

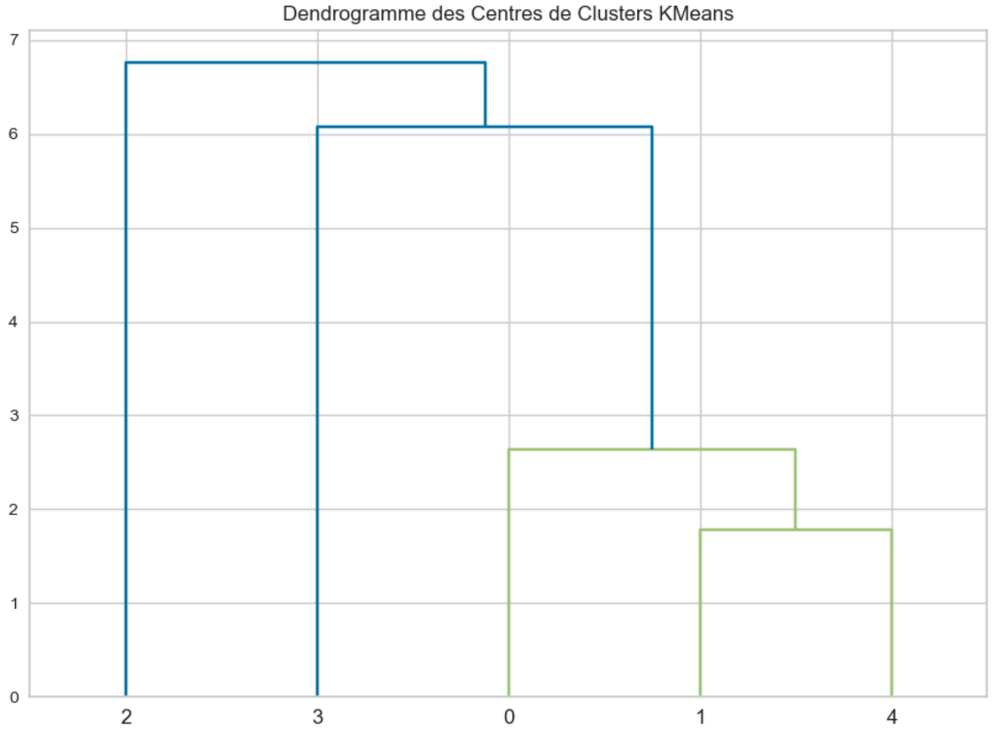
2 Clustering Hiérarchique (dendrogram) :

Complexité Computationnelle :

Le CAH a une complexité temporelle qui peut être prohibitivement élevée pour les grands ensembles de données, comme c'est souvent le cas dans le commerce en ligne. KMeans est généralement plus rapide et donc plus adapté à la manipulation de grands volumes de données.

Manque de Flexibilité :

pour l'Affectation des Clusters Une fois que les clusters sont formés dans le CAH, il n'est pas possible de réassigner les points à d'autres clusters sans reconstruire tout l'arbre hiérarchique, tandis que KMeans offre une plus grande flexibilité en permettant des réaffectations à chaque itération.



Entraînement des modèles

3 DBSCAN :

Le DBSCAN semble ne pas fonctionner efficacement sur ce jeu de données.

Cela est indiqué par le fait que tous les points sont assignés au **(label -1)** avec les paramètres initiaux **(eps=0.5 et min_samples=5)**,
et ce résultat persiste même après ajustement des paramètres.

Les scores de silhouette négatifs ou très bas suggèrent une mauvaise séparation des clusters.

Ajustement des Paramètres du DBSCAN

Les paramètres eps et min_samples de DBSCAN sont critiques et peuvent grandement influencer les résultats.

```python
# Variations des paramètres eps et min_samples
eps_values = [0.5, 1.0, 1.5]  # Liste des valeurs d'eps à tester
min_samples_values = [5, 10, 15]  # Liste des valeurs de min_samples à tester

# Expérimentation avec différents paramètres pour optimiser DBSCAN
for eps in eps_values:
    for min_samples in min_samples_values:
        dbscan = DBSCAN(eps=eps, min_samples=min_samples)
        dbscan_clusters = dbscan.fit_predict(customer_file[['recence', 'frequence', 'montant_total
        dbscan_silhouette_score = silhouette_score(customer_file[['recence', 'frequence', 'montant

        # Affichage des résultats pour chaque configuration
        print(f"DBSCAN eps={eps}, min_samples={min_samples}, clusters : {dbscan_clusters}")
        print(f"DBSCAN eps={eps}, min_samples={min_samples}, silhouette : {dbscan_silhouette_score

```

```
DBSCAN eps=0.5, min_samples=5, clusters : [-1 -1 -1 ... -1 -1 -1]
DBSCAN eps=0.5, min_samples=5, silhouette : -0.7508290222028584
DBSCAN eps=0.5, min_samples=10, clusters : [-1 -1 -1 ... -1 -1 -1]
DBSCAN eps=0.5, min_samples=10, silhouette : -0.7350282050280227
DBSCAN eps=0.5, min_samples=15, clusters : [-1 -1 -1 ... -1 -1 -1]
DBSCAN eps=0.5, min_samples=15, silhouette : -0.07575565177913743
DBSCAN eps=1.0, min_samples=5, clusters : [ 0 -1 -1 ... -1 -1 -1]
DBSCAN eps=1.0, min_samples=5, silhouette : -0.5053894146315446
DBSCAN eps=1.0, min_samples=10, clusters : [-1 -1 -1 ... -1 -1 -1]
DBSCAN eps=1.0, min_samples=10, silhouette : -0.7131242701808108
DBSCAN eps=1.0, min_samples=15, clusters : [-1 -1 -1 ... -1 -1 -1]
DBSCAN eps=1.0, min_samples=15, silhouette : -0.7402917681676412
DBSCAN eps=1.5, min_samples=5, clusters : [   0    1  576 ...  197  822 1461]
DBSCAN eps=1.5, min_samples=5, silhouette : -0.18403258218122098
DBSCAN eps=1.5, min_samples=10, clusters : [  0   1  -1 ...  -1 973  -1]
DBSCAN eps=1.5, min_samples=10, silhouette : -0.2684519682167817
DBSCAN eps=1.5, min_samples=15, clusters : [423 685  -1 ...  -1  -1  -1]
DBSCAN eps=1.5, min_samples=15, silhouette : -0.45245890693321156
```

Modèle Séléctionné

1 **K-Means :**

Application du KMeans avec le Nombre Optimal de Clusters

Suite à notre évaluation quantitative, nous avons déterminé que le nombre optimal de clusters pour notre modèle est de 5. Nous allons maintenant appliquer le modèle KMeans avec ce nombre de clusters pour segmenter nos clients.

```
# Création de l'objet KMeans avec le nombre optimal de clusters
kmeans = KMeans(n_clusters=5, random_state=42)

# Adapter les données au modèle KMeans
kmeans.fit(data)

# Assigner chaque client à un cluster
customer_file['cluster'] = kmeans.labels_
scaled_df['cluster'] = kmeans.labels_
```

Visualisation des Segments de Clients

Enfin, nous examinons les premiers clients de chaque segment pour avoir un aperçu des groupes formés.

```
customer_file.head()
```

	customer_unique_id	recence	frequence	montant_total	score_review_minimum	cluster
0	0000366f3b9a7992bf8c76cfdf3221e2	153.132905	1	129.90	5	4
1	0000b849f77a49e4a4ce2b2a4ca5be3f	159.248125	1	18.90	4	4
2	0000f46a3911fa3c0805444483337064	559.389734	1	69.00	3	1
3	0000f6ccb0745a6a4b88665a16c9f078	349.108970	1	25.99	4	1
4	0004aac84e0df4da2b147fca70cf8255	323.035463	1	180.00	5	1

Évaluation du Modèle Selectionné

Introduction aux Métriques :

```python
# Calcul et affichage des métriques de qualité des clusters
davies_bouldin = davies_bouldin_score(data, kmeans.labels_)
silhouette = silhouette_score(data, kmeans.labels_)
print(f"Score Davies-Bouldin : {davies_bouldin}")
print(f"Coefficient de silhouette : {silhouette}")
```

1. **Silhouette Score :** Le Silhouette Score est une métrique qui mesure à quel point chaque point dans un cluster est proche des points dans son propre cluster et éloigné des points des autres clusters. Le score est compris entre -1 et +1, où une valeur élevée indique que les points sont bien regroupés et bien séparés des autres clusters. Un score élevé signifie donc que la formation des clusters est cohérente et distincte.

```
Coefficient de silhouette : 0.42133394100335736
```

2. **Davies-Bouldin Index :** Le Davies-Bouldin Index est une autre métrique d'évaluation de la qualité des clusters qui repose sur le rapport entre la dispersion au sein des clusters et la séparation entre eux. Plus précisément, il mesure la moyenne de la "similarité" entre chaque cluster avec son cluster le plus similaire, où la similarité est le rapport entre les distances intra-clusters et inter-clusters. Contrairement au Silhouette Score, pour le Davies-Bouldin Index, des valeurs plus basses indiquent de meilleurs clusters, avec un score minimum de zéro indiquant une séparation optimale.

```
Score Davies-Bouldin : 0.7738038752669968
```

Évaluation du Modèle Selectionné

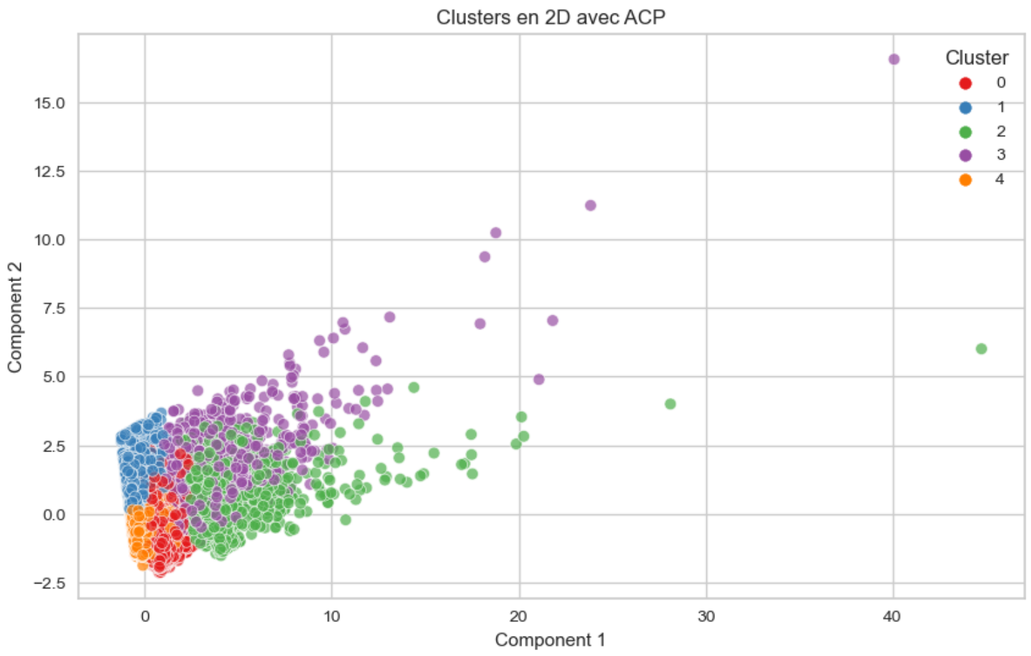
Introduction aux Métriques :

3

ACP : L'Analyse en Composantes Principales, est une technique de réduction de dimension qui nous permet de simplifier la complexité des données en les projetant sur deux dimensions principales, rendant ainsi les patterns plus faciles à visualiser et à interpréter. Sur ce graphique, chaque point représente un client, et les couleurs différentes indiquent les groupes de clients, ou "clusters", identifiés par notre modèle.

Ce que nous voyons ici :

- **Clarté de la séparation :** Les groupes sont visuellement distincts les uns des autres
- **Concentration des clusters :** La majorité des clients sont regroupés dans un espace restreint, ce qui suggère des comportements d'achat similaires au sein de ces groupes.
- **Clients isolés :** Il y a quelques points éloignés du gros du groupe, ce qui peut indiquer des clients aux comportements atypiques.



Analyse de Stabilité et Maintenance du Modèle

1 **Méthode d'Analyse de Stabilité:**

- **Adjusted Rand Index (ARI) :** Utilisé pour comparer la similitude entre deux distributions de clusters. Un score proche de 1 indique une parfaite stabilité, tandis qu'un score proche de 0 révèle une instabilité significative.
- **Importance de l'ARI :** Un indicateur fiable pour déterminer si le comportement d'achat des clients a changé au point de nécessiter un réentraînement du modèle de clustering.

2 **Fréquence de Réentraînement:**

- **Basée sur la volatilité des scores ARI :** Des variations importantes suggèrent une évolution des patterns de clients et la nécessité de mettre à jour le modèle.
- **Règle générale :** Réentraînement envisagé si le score ARI chute en dessous d'un seuil prédéfini (0.80).

3 **Plan de Maintenance :**

- **Surveillance Régulière :** Suivi hebdomadaire des scores ARI pour détecter des changements dans les comportements de clients.
- **Mise à jour du Modèle :** Si des changements sont détectés, le modèle est réentraîné avec de nouvelles données.

```
Semaine 1: ARI =
Semaine 2: ARI =
Semaine 3: ARI =
Semaine 4: ARI =
Semaine 5: ARI =
Semaine 6: ARI =
Semaine 7: ARI =
Semaine 8: ARI =
Semaine 9: ARI =
Semaine 10: ARI =
Semaine 11: ARI =
Semaine 12: ARI =
```

Analyse de Stabilité et Maintenance du Modèle

3

Plan de Maintenance :

Mise à jour du Modèle :

- **Semaines 1 à 3 :** Durant les premières trois semaines, les scores ARI demeurent élevés, atteignant 0.97 au départ, signe d'une grande stabilité des clusters initiaux malgré l'ajout de nouvelles données.
- **Semaines 4 à 8 :** Entre la 4e et la 8e semaine, une baisse claire des scores ARI, de 0.886 à 0.800, révèle une divergence croissante entre les clusters existants et ceux formés avec les nouvelles données.
- **Semaines 9 à 12 :** Dans les dernières semaines, de 9 à 12, le score ARI diminue jusqu'à 0.721, signalant une divergence croissante des clusters au fil du temps et suggérant une instabilité face aux nouvelles données.

```
Semaine 1: ARI = 0.9714758247478267
Semaine 2: ARI = 0.9442954284155339
Semaine 3: ARI = 0.9198409454785712
Semaine 4: ARI = 0.8911151710580039
Semaine 5: ARI = 0.8842880190152195
Semaine 6: ARI = 0.8692858369370704
Semaine 7: ARI = 0.8477718232524482
Semaine 8: ARI = 0.803736544632822
Semaine 9: ARI = 0.7914657313857845
Semaine 10: ARI = 0.7595880943039054
Semaine 11: ARI = 0.7453238078469516
Semaine 12: ARI = 0.7283947438271764
```

Analyse de Stabilité et Maintenance du Modèle

3

Plan de Maintenance :

Mise à jour du Modèle :

Conclusion : Points Clés d'Intervention pour le Réentraînement

- Après une Chute Notable de l'ARI : Un bon indicateur pour le réentraînement peut être identifié lorsque l'**on observe une chute notable du score ARI d'une semaine à l'autre.** Par exemple, entre la **semaine 7 et la semaine 8, le score ARI passe de 0.841 à 0.800.** Ce point marque une divergence croissante et pourrait être un moment opportun pour réentraîner le modèle afin de réaligner les clusters avec les tendances actuelles des données.

```
Semaine 1: ARI = 0.9714758247478267
Semaine 2: ARI = 0.9442954284155339
Semaine 3: ARI = 0.9198409454785712
Semaine 4: ARI = 0.8911151710580039
Semaine 5: ARI = 0.8842880190152195
Semaine 6: ARI = 0.8692858369370704
Semaine 7: ARI = 0.8477718232524482
Semaine 8: ARI = 0.803736544632822
Semaine 9: ARI = 0.7914657313857845
Semaine 10: ARI = 0.7595880943039054
Semaine 11: ARI = 0.7453238078469516
Semaine 12: ARI = 0.7283947438271764
```

Merci

Avez-vous des question ?

www.olist.com