
# Analyser les ventes d'une librairie avec Python

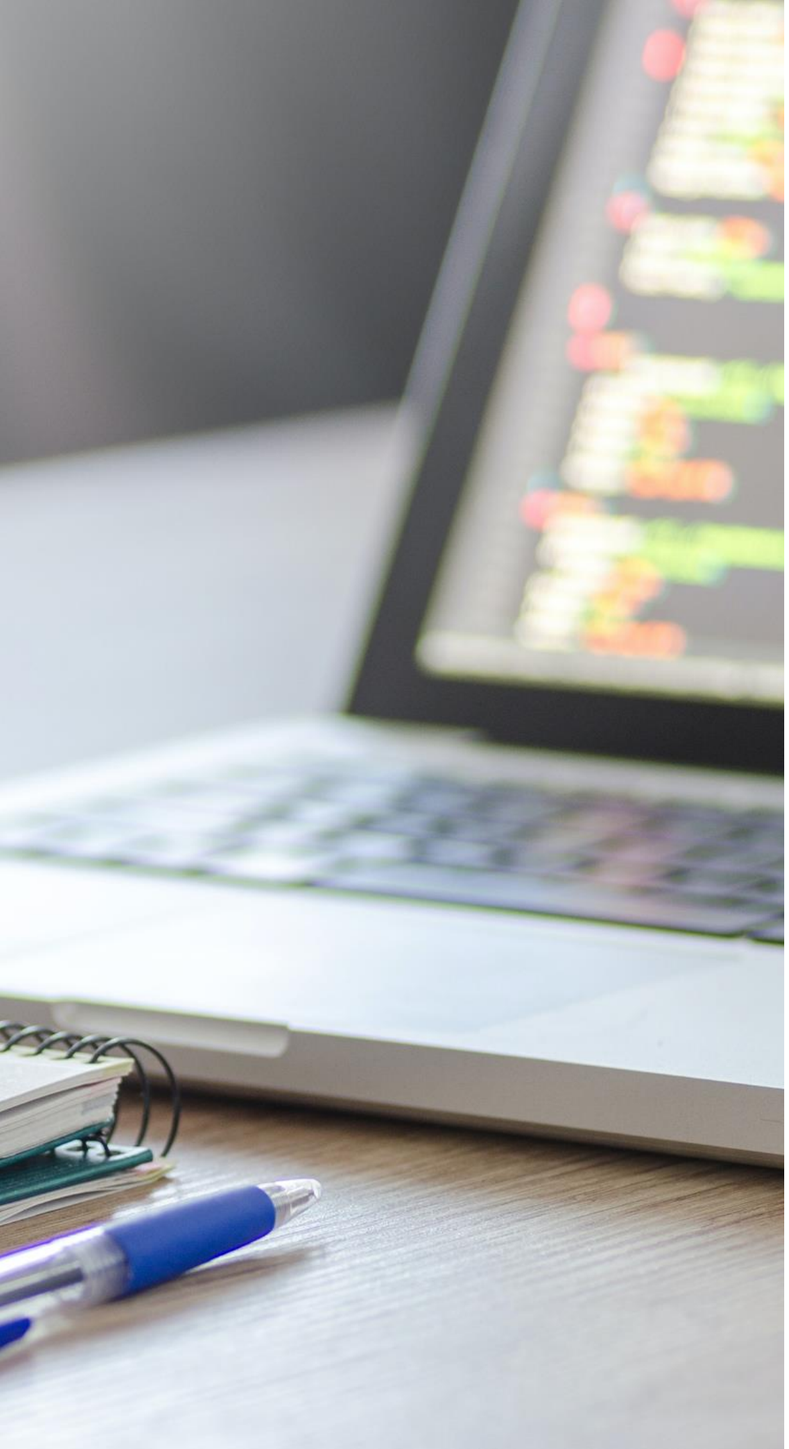
Lapage

Lapage était originellement une librairie physique avec plusieurs points de vente.
Mais devant le succès de certains de ses produits et l'engouement de ses clients, elle a décidé depuis 2 ans d'ouvrir un site de vente en ligne.
L'entreprise souhaite faire le point après deux ans d'exercice et pouvoir analyser ses points forts, ses points faibles, les comportements clients, etc.

Pour cela, nous avons à notre disposition 3 bases de données :
1) "customers" (identifiant client, la date de naissance et le genre du client)
2) "produtcs" (identifiant produit, prix du livre et la catégorie à laquelle appartient le livre)
3) "transactions" ( identifiant client, identifiant produit, identifiant de la session et la date)

<u>Les requêtes sont les suivantes</u> :

- L'évolution dans le temps et la mise en place d'une décomposition en moyenne mobile pour évaluer la tendance globale
- La répartition du CA entre les clients via une courbe de Lorenz
- Concernant le CA : la répartition par catégorie, genre ou encore année
- Tops et flops (référence)
- Le lien entre le genre d'un client et les catégories des livres achetés
- Le lien entre l'âge des clients et le montant total des achats
- Le lien entre l'âge et la fréquence d'achat
- Le lien entre l'âge et la taille du panier moyen
- Le lien entre l'âge et les catégories des livres achetés



# TRAITEMENT DES TABLES

Les tables "customers" et "products" n'avaient pas besoin de traitement particulier (vérification de l'unicité), a contrario de la base "transactions".
Des phases test ont été effectuées et se sont retrouvées dans la base sur plusieurs lignes (200).
Elles se démarquent par la présence du mot "test" et la date est identifique (2021-03-01),
l'identifiant du produit est noté "T_0" et l'identifiant client est soit "ct_0" ou "ct_

| | | | | |
|---|---|---|---|---|
| **657830** | T_0 | test_2021-03-01 02:30:02.237417 | s_0 | ct_0 |
| **662081** | T 0 | test 2021-03-01 02:30:02.237427 | s 0 | ct 1 |

Ainsi, je supprime les 200 lignes liées à la phase test.

```
transactions.drop(transactions.loc[transactions['id_prod']=="T_0"].index, inplace=True)
```

De plus, je supprime également les 2 lignes rattachées aux identifiants client "ct_0" et "ct_1".

```
customers.drop(customers.loc[customers['client_id']=="ct_0"].index, inplace=True)
```

```
customers.drop(customers.loc[customers['client_id']=="ct_1"].index, inplace=True)
```

Enfin, je supprime la ligne rattachée à l'identifiant produit "T_0" (prix négatif).

| | id_prod | price | categ |
|---|---|---|---|
| **731** | T_0 | -1.0 | 0 |

```
products.drop(products.loc[products["id_prod"]=="T_0"].index, inplace=True)
```

# Jointure (1/2)

Une fois le traitement des différentes tables effectué, j'effectue deux jointures afin d'avoir une table complète comprenant l'ensemble des informations.

<u>1ère jointure</u> : Entre la table "transactions" et "products" avec "id_prod" comme clef unique. Je nomme cette table "merge1".

```
merge1 = pd.merge(transactions, products, on='id_prod', how='outer', indicator=True)
merge1
```

| | id_prod | date | session_id | client_id | price | categ | _merge |
|---|---|---|---|---|---|---|---|
| **0** | 0_1518 | 2022-05-20 13:21:29.043970 | s_211425 | c_103 | 4.18 | 0.0 | both |
| **1** | 0_1518 | 2021-09-26 12:37:29.780414 | s_95811 | c_6197 | 4.18 | 0.0 | both |
| **2** | 0_1518 | 2021-05-06 17:14:43.117440 | s_30782 | c_682 | 4.18 | 0.0 | both |
| **3** | 0_1518 | 2022-03-16 18:57:10.420103 | s_180057 | c_5932 | 4.18 | 0.0 | both |
| **4** | 0_1518 | 2022-11-12 18:58:10.574853 | s_296584 | c_7217 | 4.18 | 0.0 | both |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **679348** | 0_1624 | NaN | NaN | NaN | 24.50 | 0.0 | right_only |
| **679349** | 2_86 | NaN | NaN | NaN | 132.36 | 2.0 | right_only |
| **679350** | 0_299 | NaN | NaN | NaN | 22.99 | 0.0 | right_only |
| **679351** | 0_510 | NaN | NaN | NaN | 23.66 | 0.0 | right_only |
| **679352** | 0_2308 | NaN | NaN | NaN | 20.28 | 0.0 | right_only |

Les 21 lignes ("right_only" dans la colonne "_merge") ne possèdent pas d'identifiant client et sont supprimées (car inexploitables).

# Jointure (2/2)

**2ème jointure** : Entre la table "merge1" et "customers" avec "client_id" comme clef unique. Je nomme cette table "merge2".

```
merge2 = pd.merge(merge1, customers, on="client_id", how="inner")
merge2
```

| | id_prod | date | session_id | client_id | price | categ | _merge1 | sex | birth |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 0_1518 | 2022-05-20 13:21:29.043970 | s_211425 | c_103 | 4.18 | 0.0 | both | f | 1986 |
| **1** | 0_1518 | 2021-07-20 13:21:29.043970 | s_64849 | c_103 | 4.18 | 0.0 | both | f | 1986 |
| **2** | 0_1518 | 2022-08-20 13:21:29.043970 | s_255965 | c_103 | 4.18 | 0.0 | both | f | 1986 |
| **3** | 0_1418 | 2022-06-18 01:49:37.823274 | s_225411 | c_103 | 8.57 | 0.0 | both | f | 1986 |
| **4** | 0_1418 | 2021-08-18 01:49:37.823274 | s_77214 | c_103 | 8.57 | 0.0 | both | f | 1986 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **679327** | 2_147 | 2021-07-23 07:01:38.963669 | s_65994 | c_4391 | 181.99 | 2.0 | both | f | 2001 |
| **679328** | 0_142 | 2022-03-25 18:07:25.880052 | s_184472 | c_1232 | 19.85 | 0.0 | both | f | 1960 |
| **679329** | 0_142 | 2021-09-25 18:07:25.880052 | s_95415 | c_1232 | 19.85 | 0.0 | both | f | 1960 |
| **679330** | 2_205 | 2021-03-11 00:11:32.016264 | s_4605 | c_7534 | 100.99 | 2.0 | both | m | 1992 |
| **679331** | 2_205 | 2022-07-11 00:11:32.016264 | s_236636 | c_7534 | 100.99 | 2.0 | both | m | 1992 |

679332 rows × 9 columns

Je vérifie le format des données et après vérification, je convertis la colonne "date" au format datetime.

```
merge2['date'] = pd.to_datetime(merge2['date'])
```

# Traitement de merge2

Dans les requêtes, il est demandé d'étudier l'évolution du CA et de le visualiser en effectuant la moyenne mobile et une courbe de Lorenz.
Ainsi, pour la moyenne mobile j'ai besoin d'une colonne avec l'année et mois tandis que pour la courbe de Lorenz, j'ai besoin d'une colonne avec l'année, le mois et le jour.
Je crée donc 2 colonne dans ce sens

**Colonne année et mois :**

```
merge2['ym-date'] = merge2['date'].dt.strftime('%Y-%m')
```

```
merge2.head(2)
```

| | id_prod | date | session_id | client_id | price | categ | _merge1 | sex | birth | ym-date |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0_1518 | 2022-05-20 13:21:29.043970 | s_211425 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2022-05 |
| **1** | 0_1518 | 2021-07-20 13:21:29.043970 | s_64849 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2021-07 |

**Colonne année, mois et jour (sans l'heure) :**

```
merge2['periode'] = merge2['date'].dt.date
merge2['temps'] = merge2['date'].dt.time
```

```
merge2.head(2)
```

| | id_prod | date | session_id | client_id | price | categ | _merge1 | sex | birth | ym-date | periode | temps |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0_1518 | 2022-05-20 13:21:29.043970 | s_211425 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2022-05 | 2022-05-20 | 13:21:29.043970 |
| **1** | 0_1518 | 2021-07-20 13:21:29.043970 | s_64849 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2021-07 | 2021-07-20 | 13:21:29.043970 |

Il sera également demandé des requêtes incluant l'âge. Je crée également une colonne dans ce sens.

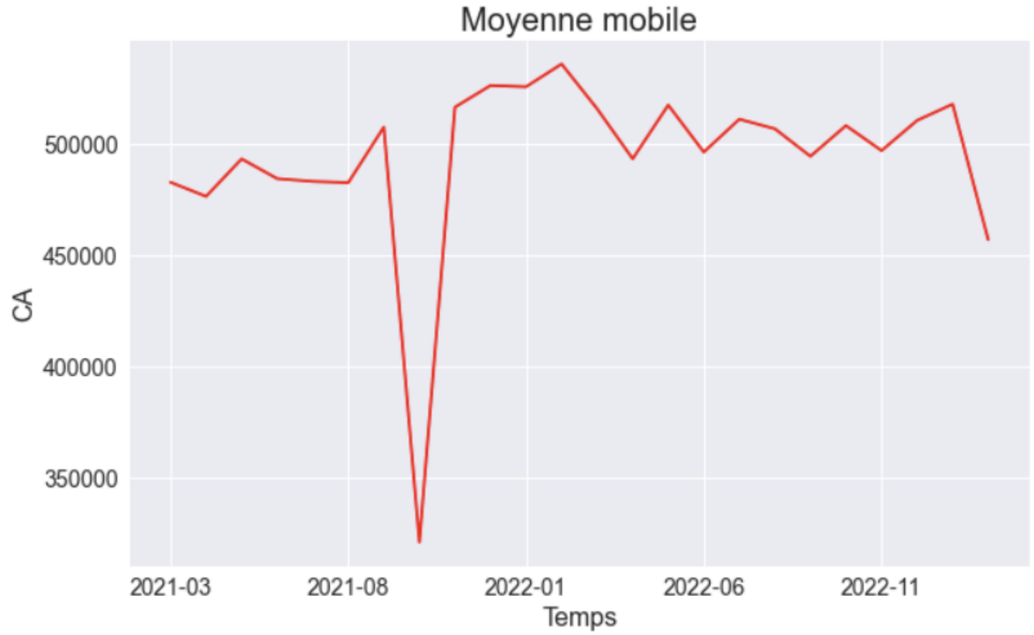
```
merge2['age'] = 2022- merge2['birth']
```

```
merge2.head(2)
```

| | id_prod | date | session_id | client_id | price | categ | _merge1 | sex | birth | ym-date | periode | temps | Year | Month | age |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0_1518 | 2022-05-20 13:21:29.043970 | s_211425 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2022-05 | 2022-05-20 | 13:21:29.043970 | 2022 | 5 | 36 |
| **1** | 0_1518 | 2021-07-20 13:21:29.043970 | s_64849 | c_103 | 4.18 | 0.0 | both | f | 1986 | 2021-07 | 2021-07-20 | 13:21:29.043970 | 2021 | 7 | 36 |

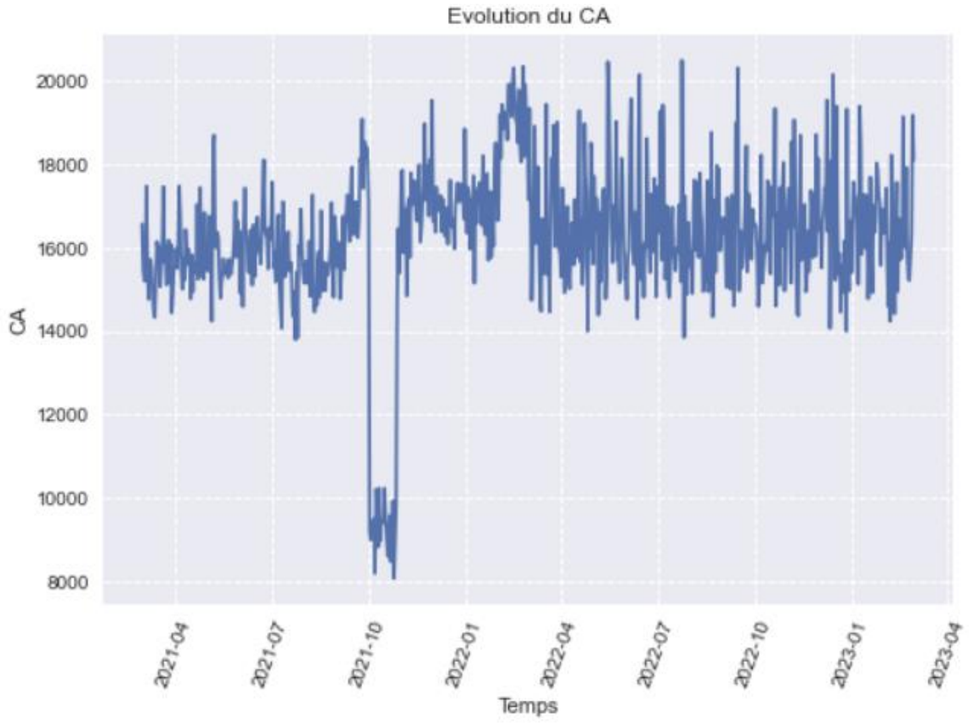
## Moyenne mobile

**Baisse significative du CA en octobre 2021**




## Évolution du CA

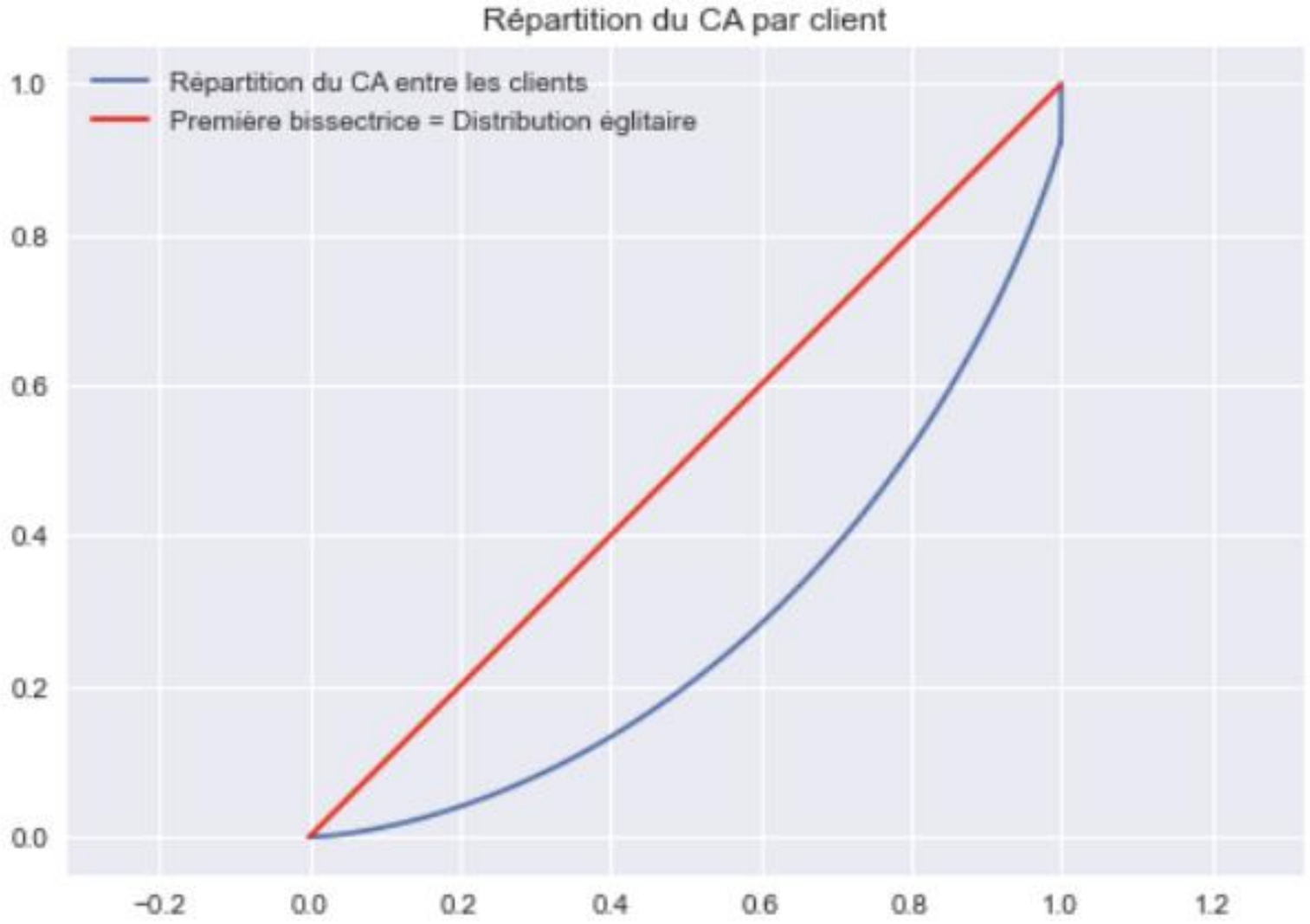
**Confirme la tendance du graphique précédent à savoir la baisse du CA au mois d'octobre 2021**



# Répartition du CA par client (courbe de Lorenz)

- **Plus la courbe de Lorenz est proche de la première bissectrice, plus la répartition est égalitaire. Ce qui n'est pas le cas ici.**

- L'indice de GINI mesure le niveau d'inégalité de la répartition d'une variable dans la population. Le coefficient varie de 0 (égalité parfaite) à 1 (inégalité parfaite. Dans notre cas, l'indice de GINI est de 0.45




```
AUC = (lorenz.sum() -lorenz[-1]/2 -lorenz[0]/2)/n
S = 0.5 - AUC # surface entre la première bissectr
gini = 2*S
gini
```

```
0.44638654137401435
```

# A) Le lien entre l'âge des clients et le montant total des achats

- Les femmes semblent acheter plus de livres appartenant à la catégorie 1.

- Tandis que les hommes semblent acheter légèrement plus de livres appartenant à la catégorie 0 et 2

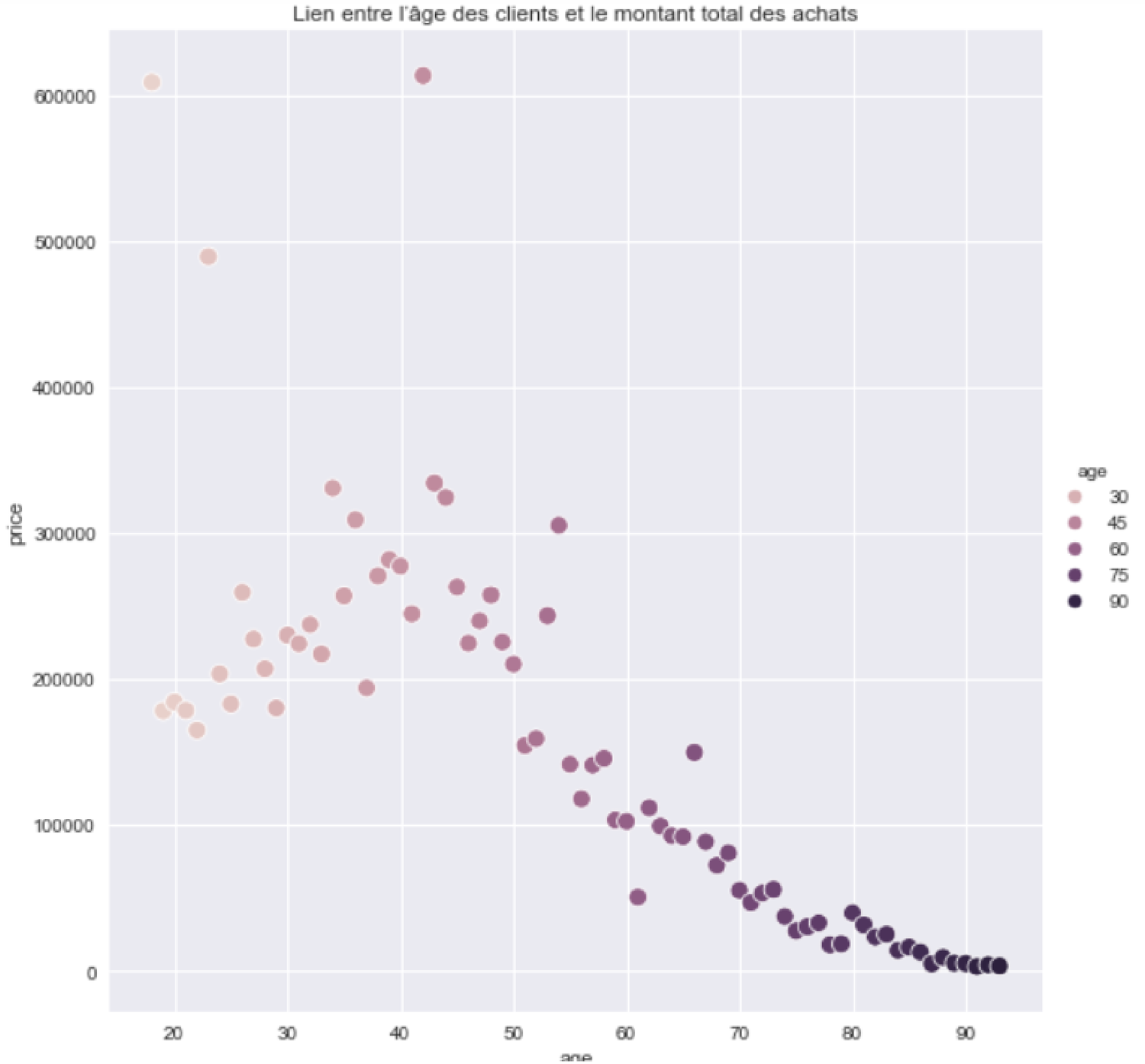
# B) Le lien entre le genre d'un client et le montant total des achats

- Il semble que plus les clientssoientt âgés, moins le montant total des achats soit important.

- Il semble que la catégorie des 30-45 soit la tranche d'âge où le coup total est le plus important.

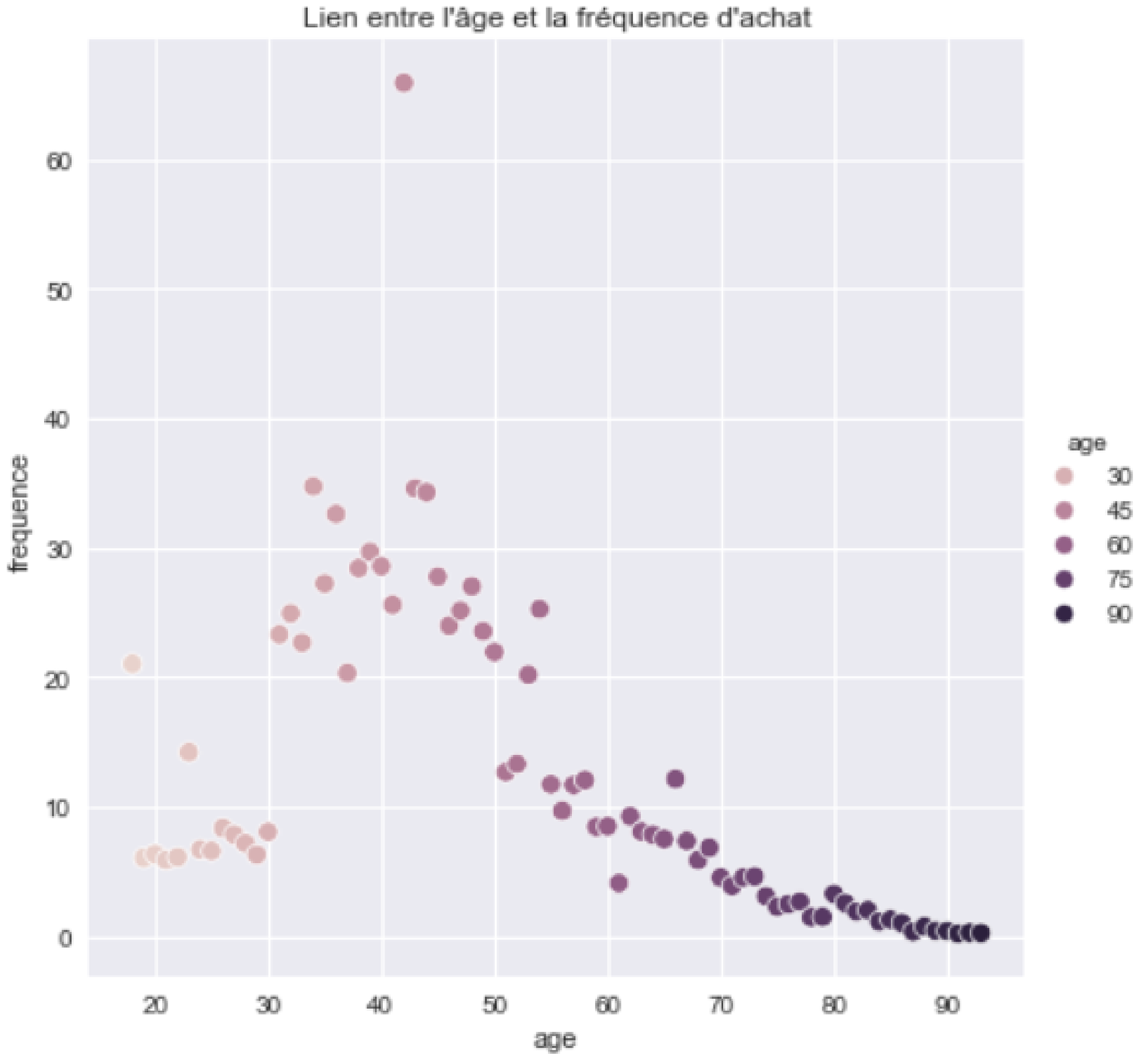
- On notera toutefois que la catégorie des 18 ans est celle faisant le plus d'achat, juste après la catégorie des 42 ans.

- **Cette surepresentation de la catégorie des "18 ans" peut s'expliquer car il faut fournir un âge minimum pour acheter sur un site internet : il faut être majeur (18 ans).**



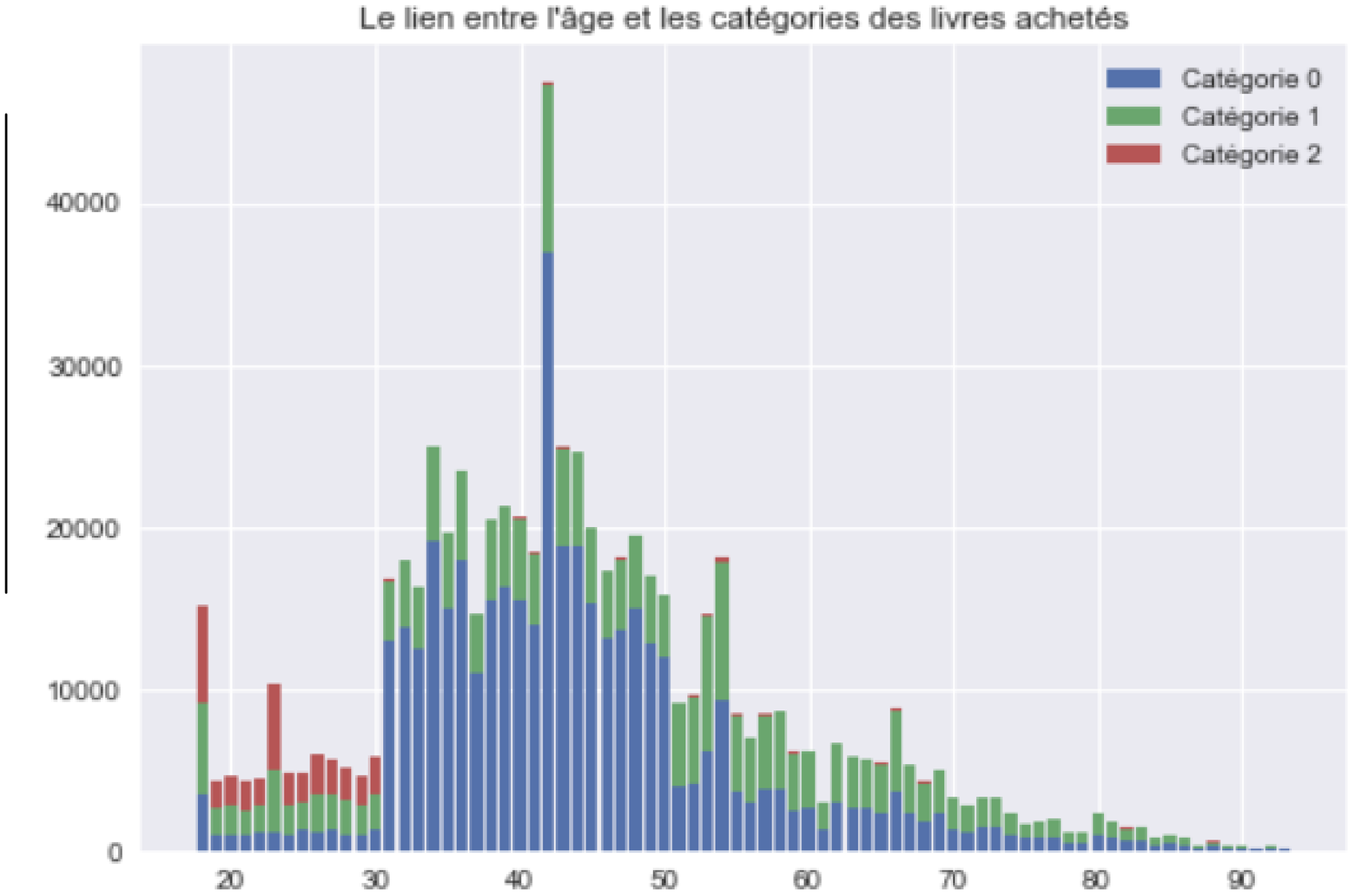
## C) Le lien entre l'âge et la fréquence d'achat

- Les 30-40 ans sont la catégorie avec la fréquence d'achat la plus élevée.
- Avec un pic notable pour la catégorie des 42 ans.
- A partir de 50 ans, plus les clients sont âgés, plus la fréquence d'achat est faible.



# D) Le lien entre l'âge et les catégories des livres achetés

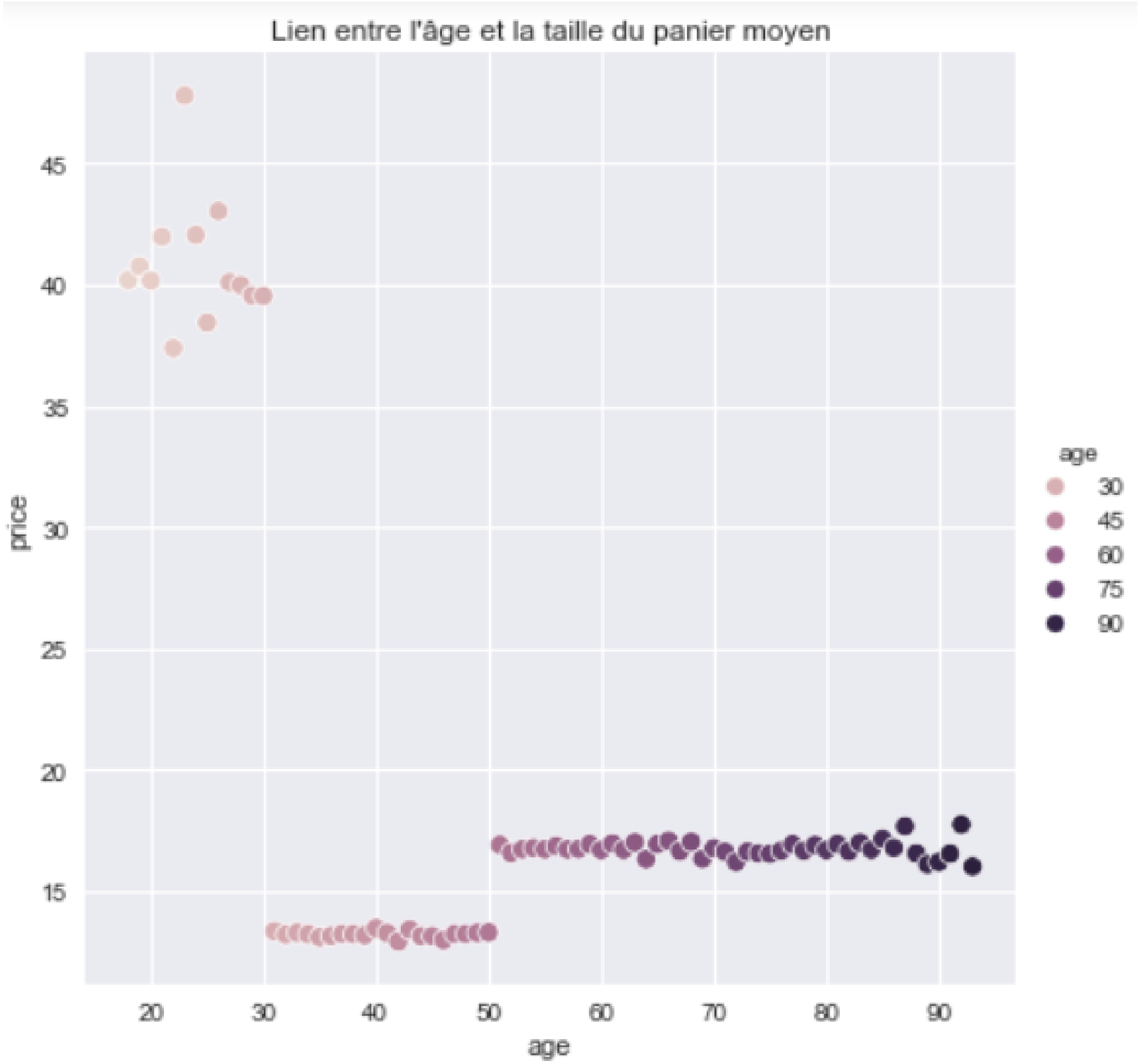
- Les 18-30 ans sont la catégorie achetant le plus de livres appartenant à la catégorie 2 (rouge).
- Les 31-55 sont la catégorie achetant le plus de livres appartenant à la catégorie 0 (bleu).
- Les livres de catégorie 1 (vert) sont achetés par l'ensemble des catégories. On notera toutefois que les plus de 70 ans semblent acheter en majorité des livres issus de cette catégorie.



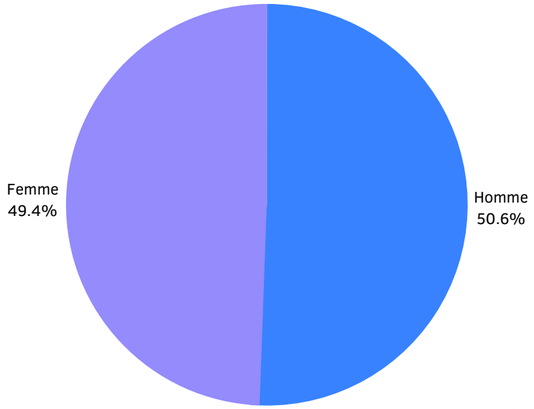
## E) Le lien entre l'âge et la taille du panier moyen

3 tendances semblent se démarquer :

- Les - 30 ans ont le panier moyen le plus important (compris entre 37 et 48 euros).
- Les clients entre 30 ans et 50 ans ont un panier moyen de 13 euros.
- Enfin, les clients en 50 et 93 ans ont un panier moyen entre 16 euros et 25 euros).

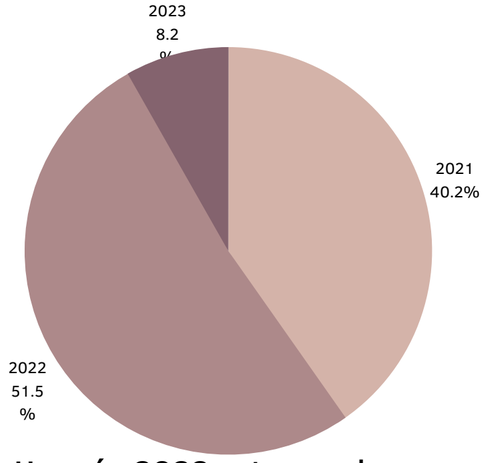


## Répartition du CA en fonction du genre



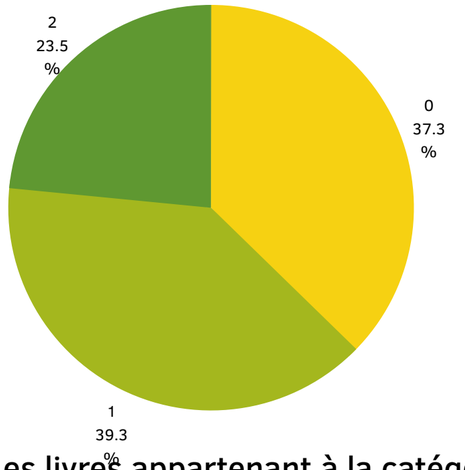

Les hommes sont légèrement plus nombreux sur le site.

## Répartition du CA en fonction de l'année



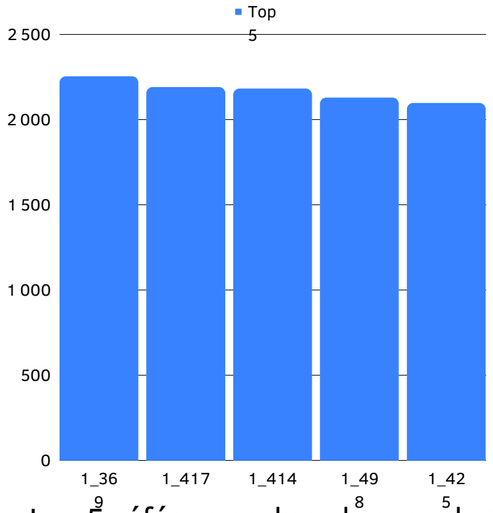

L'année 2022 est, pour le moment, l'année avec le meilleur CA

## Répartition du CA en fonction de la catégorie des livres achetés

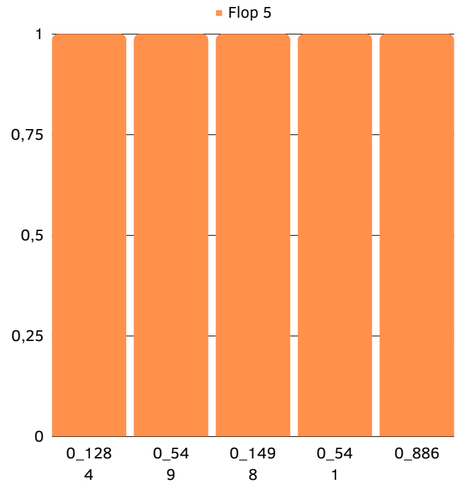



Les livres appartenant à la catégorie 1 sont les plus vendus (cela concorde avec les observations du graphique "Le lien entre l'âge et les catégories des livres achetés")

## Tops et flops (référence)

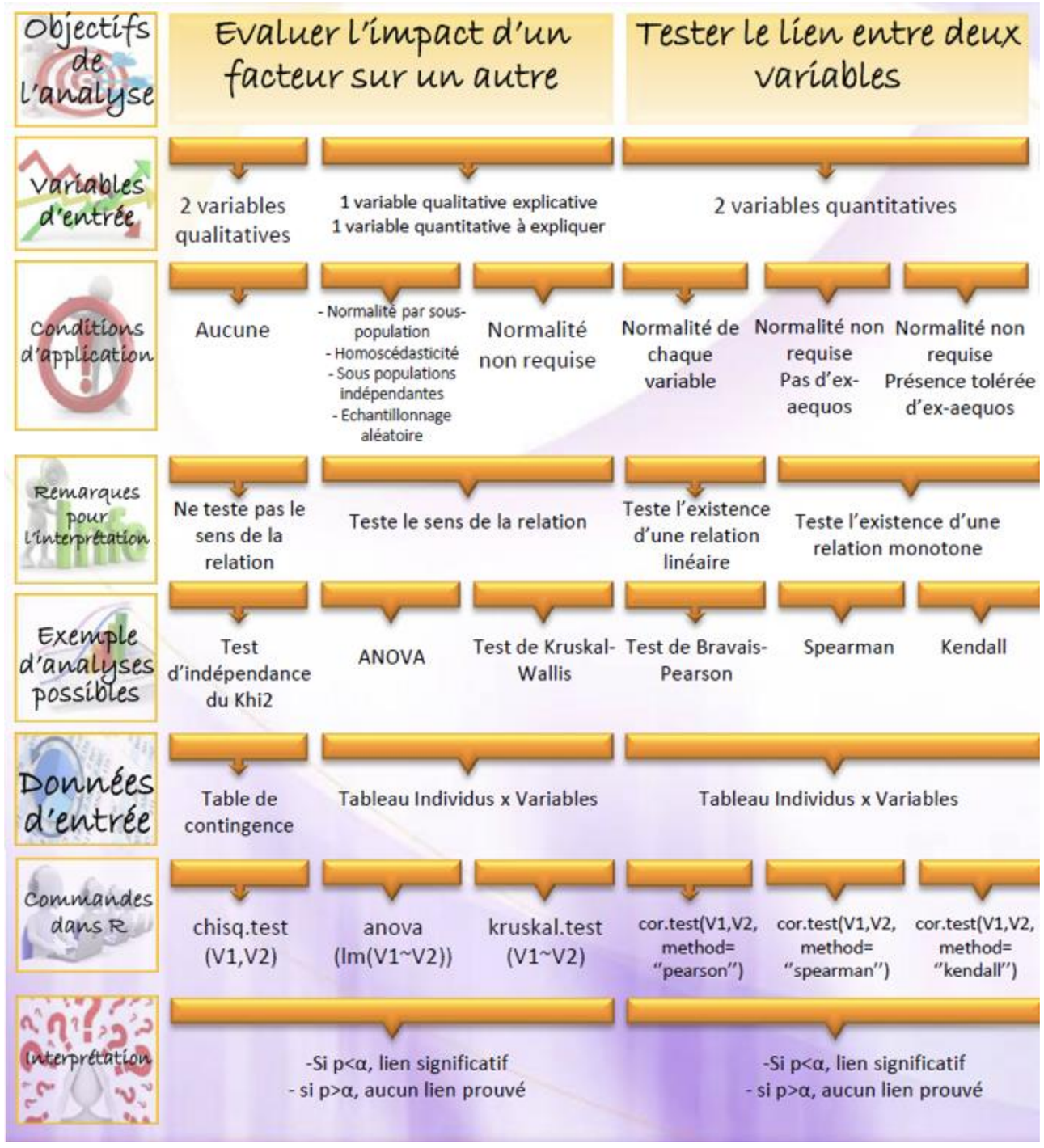



Les 5 références les plus vendues appartiennent à la catégorie 1




Les 5  références les moins vendues appartiennent à la catégorie 0 (seulement 1 unité vendue).

# Test statistique (1/2)




Un test statistique est une procédure de décision entre 2 hypothèses. Il s'agit d'une démarche consistant à rejeter ou à ne pas rejeter une hypothèse statistique, appelée hypothèse nulle, en fonction d'un échantillon de données.

Dans notre cas, il s'agit de tester un éventuel lien entre deux variables, en fonction des observations faites.

Afin de choisir le test adéquat, nous devons :
1) Déterminer la nature des variables (quantitatives, qualitatives)
2) Tester la normalité (si la distribution de la variable suit une loi normale, on effectuera un test paramétrique. A contrario, ce sera un test non paramétrique).

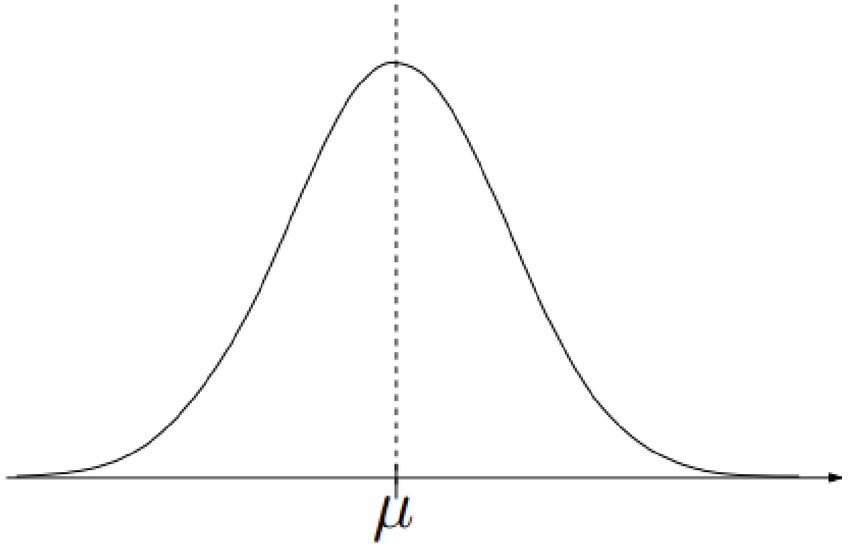
# Test statistique (2/2)

- Les variables quantitatives sont des variables pouvant se traduire par des valeurs numériques.

- Les variables qualitatives sont des variables caractérisant l'appartenance de l'individu à un groupe (ou une catégorie).

- La loi normale définit une représentation de données selon laquelle la plupart des valeurs sont regroupées autour de la moyenne et les autres s'en écartent symétriquement des deux côtés. La loi normale est la loi des phénomènes naturels car elle est très répandue, entre autre, dans le domaine des sciences naturelles (géologie, biologie, ...).

- La représentation graphique d'une loi normale est parfois appelée courbe en cloche. La forme exacte varie selon la répartition de la population, mais le sommet est toujours situé au milieu et la courbe est toujours symétrique. La moyenne, le mode et la médiane d'une loi normale sont identiques.




- Courbe symétrique par rapport μ
- Forme de cloche

# A) Test statistique entre le genre d'un client et les catégories des livres achetés

- Nous avons 2 variables qualitatives à savoir le genre (h/f) et la catégorie de livre (3 catégories).
- Nous utilisons donc un **test de CHI2.**
- Ce test permet de vérifier l'absence de lien statistique entre deux variables X et Y. Les deux sont dites indépendantes lorsqu'il n'existe aucun lien statistique entre elles, dit autrement, la connaissance de X ne permet en aucune manière de se prononcer sur Y. L'hypothèse nulle (H0) de ce test est la suivante : les deux variables X et Y sont indépendantes.
- En termes de valeur p, l'hypothèse nulle est généralement rejetée lorsque $p \leq 0,05$.

- **Existe t-il un lien entre le genre d'un client et la catégorie des livres achetés ?**

H0 = Il ne semble pas exister de lien entre le genre du client et la catégorie de livre acheté

```
X = "sex"
Y = "categ"
cont = merge2[[X, Y]].pivot_table(index=X, columns=Y, aggfunc=len).fillna(0).copy().astype(int)
cont
```

| categ | 0.0 | 1.0 | 2.0 |
|---|---|---|---|
| sex | | | |
| f | 206103 | 114899 | 17283 |
| m | 209356 | 112270 | 19200 |

```
from scipy.stats import chi2_contingency as chi2_contingency

khi2, pval , ddl , contingent_theorique = chi2_contingency(cont)
```

```
print(pval)
```

```
1.1310980597090762e-32
```

Conclusion :

Nous constatons que la pvalue est inférieure à 0.05 donc nous pouvons rejetter l'hypothèse nulle.

**Ainsi, il semble exister un lien entre le genre d'un client et la catégorie des livres achetés.**

# B) Test statistique entre l'âge des clients et le montant total des achats

- Nous avons 2 variables quantitatives à savoir l'âge des clients et le montant total des achats
- Pour les variables quantitatives, la question préalable à se poser est de savoir si la distribution des variables suit une loi normale ou non. Pour cela, il faut utiliser le test de Kolmogorov-Smirnov. Si la pvalue est inférieure à 0.05, la variable ne suit pas la loi normale.
- Dans le cas où les deux variables suivent la loi normale, nous utiliserons un test-paramétrique : Pearson
- Dans le cas contraire, si l'une des deux variables ou les deux ne suivent pas la loi normale, nous utiliserons un test non-paramétrique : Spearman

```
from numpy.random import seed
from numpy.random import randn
from numpy.random import lognormal

#set seed (e.g. make this example reproducible)
seed(0)

#generate two datasets
datax = amontant['age']
datay = amontant['price']
```

```
from scipy import stats
stats.kstest(datay, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

```
from scipy import stats
stats.kstest(datax, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

<u>Conclusion (Test de la loi normale) :</u>

L'âge ainsi que le montant total des achats ne suivent pas une loi normale car la pvalue < 0.05.

Nous utilisons donc le test de Spearman.

- **Existe t-il un lien entre l'âge d'un client et le montant total des achats ?**

- H0 = Il ne semble pas exister de lien entre l'âge d'un client et le montant total des achats
- Si pvalue < 0.05 alors H0 peut être rejettée

```
from scipy import stats
xm = amontant['age']
ym = amontant['price']
stats.spearmanr(xm,ym)
```

```
SpearmanrResult(correlation=-0.8576076555023923, pvalue=4.57972879340901e-23)
```

<u>Conclusion (Test de Spearman) :</u>

Nous constatons que la pvalue est inférieure à 0.05 donc nous pouvons rejetter l'hypothèse nulle.

Ainsi, il semble exister un lien entre l'âge d'un client et le montant total de ses achats.

# C) Test statistique entre l'âge et la fréquence d'achat

```
from numpy.random import seed
from numpy.random import randn
from numpy.random import lognormal

#set seed (e.g. make this example reproducible)
seed(0)

#generate two datasets
dataagef = frequence1['age']
datafrequence = frequence1['frequence']
```

```
from scipy import stats
stats.kstest(dataagef, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

```
from scipy import stats
stats.kstest(datafrequence, 'norm')
```

```
KstestResult(statistic=0.8153362438474665, pvalue=1.1066513349295498e-55)
```

<u>Conclusion (Test de la loi normale) :</u>

L'âge ainsi que la fréquence d'achat ne suivent pas une loi normale car la pvalue < 0.05.

Nous allons utiliser le test de Spearman.

- **Existe t-il un lien entre l'âge d'un client et la fréquence d'achat?**

- H0 = Il ne semble pas exister de lien entre l'âge d'un client et la fréquence d'achat.
- Si pvalue < 0.05 alors H0 peut être rejettée.

```
from scipy import stats
stats.spearmanr(dataagef,datafrequence)
```

```
SpearmanrResult(correlation=-0.6846206425153794, pvalue=9.152883867240306e-12)
```

<u>Conclusion (Test de Spearman) :</u>

Nous constatons que la pvalue est inférieure à 0.05 donc nous pouvons rejetter l'hypothèse nulle.

Ainsi, il semble exister un lien entre l'âge d'un client et la fréquence d'achat.

# D) Test statistique entre l'âge et les catégories des livres achetés

- Nous avons 1 variable quantitative (l'âge) et une variable qualitative (les catégories des livres achetés)
- Nous devons savoir si la distributions suit une loi normale ou non. Pour cela, il faut utiliser le test de Kolmogorov-Smirnov. Si la pvalue est inférieure à 0.05, la variable ne suit pas la loi normale.
- Dans le cas où les deux variables suivent la loi normale, nous utiliserons un test-paramétrique : Anova
- A contrario, si l'une des deux variables ou les deux ne suivent pas la loi normale, nous utiliserons un test non-paramétrique : Kruskal-Wallis

```
#set seed (e.g. make this example reproducible)
seed(0)

#generate two datasets
data1 = acateg['age']
data2 = acateg['categ']
```

```
from scipy import stats
stats.kstest(data1, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

```
from scipy import stats
stats.kstest(data2, 'norm')
```

```
KstestResult(statistic=0.5080114127352096, pvalue=4.618782230126399e-55)
```

<u>Conclusion (Test de la loi normale) :</u>

L'âge ainsi que la catégorie des livres achetés ne suivent pas une loi normale car la pvalue < 0.05.

Nous allons utiliser le Test de Kruskall-Wallis.

- **Existe t-il un lien entre l'âge et les catégories des livres achetés ?**

- H0 = Il ne semble pas exister de lien entre l'âge et la catégorie des livres achetés.
- Si pvalue <0.05 alors H0 peut etre rejettée.

```
from scipy import stats
xca = acateg['age']
yca = acateg['categ']
stats.kruskal(xca, yca)
```

```
KruskalResult(statistic=346.06393277524035, pvalue=3.049914620216614e-77)
```

<u>Conclusion (Test de Kruskall-Wallis) :</u>

Nous constatons que la pvalue est inférieure à 0.05 donc nous pouvons rejetter l'hypothèse nulle.

Ainsi, il semble exister un lien entre l'âge d'un client et la catégorie des livres achetés.

# E) Test statistique entre le lien entre l'âge et la taille du panier moyen

```
from numpy.random import seed
from numpy.random import randn
from numpy.random import lognormal

#set seed (e.g. make this example reproducible)
seed(0)

#generate two datasets
dataage = amoyen['age']
dataprice = amoyen['price']
```

```
from scipy import stats
stats.kstest(dataprice, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

```
from scipy import stats
stats.kstest(dataage, 'norm')
```

```
KstestResult(statistic=1.0, pvalue=0.0)
```

<u>Conclusion (Test de la loi normale) :</u>

L'âge ainsi que la taille du panier moyen ne suivent pas une loi normale car la pvalue. < 0.05.

Nous allons utiliser le test de Spearman.

- **Existe t-il un lien entre l'âge d'un client et la taille du panier moyen ?**

**-** H0 = Il ne semble pas exister de lien entre l'âge d'un client et la taille du panier moyen.
- Si pvalue < 0.05 alors H0 peut être rejettée.

```
from scipy import stats
stats.spearmanr(dataage,dataprice)
```

```
SpearmanrResult(correlation=-0.08650717703349281, pvalue=0.45745267914381693)
```

<u>Conclusion (test de Spearman) :</u>

Nous constatons que la pvalue est supérieure à 0.05 donc nous pouvons admettre l'hypothèse nulle (H0).

Ainsi, il ne semble pas exister un lien entre l'âge d'un client et la taille du panier moyen.