

Projet

Analyser les ventes d'une librairie avec Python

Lapage-une grande librairie généraliste en ligne

Bases de données pour l'analyse

1. products
2. customers
3. transactions

Entrée [3]: `prod=pd.read_csv("products.csv")
prod.head()`

Out[3]:

	id_prod	price	categ
0	0_1421	19.99	0
1	0_1368	5.13	0
2	0_731	17.99	0
3	1_587	4.99	1
4	0_1507	3.99	0

Entrée [4]: `custom=pd.read_csv("customers.csv")
custom.head()`

Out[4]:

	client_id	sex	birth
0	c_4410	f	1967
1	c_7839	f	1975
2	c_1699	f	1984
3	c_5961	f	1962
4	c_5320	m	1943


Entrée [5]: `trans=pd.read_csv("transactions.csv")
trans.head()`

Out[5]:

	id_prod	date	session_id	client_id
0	0_1518	2022-05-20 13:21:29.043970	s_211425	c_103
1	1_251	2022-02-02 07:55:19.149409	s_158752	c_8534
2	0_1277	2022-06-18 15:44:33.155329	s_225667	c_6714
3	2_209	2021-06-24 04:19:29.835891	s_52962	c_6941
4	0_1509	2023-01-11 08:22:08.194479	s_325227	c_4232

Nettoyage le jeu de données

- Produits de test : J'ai trouvée et supprimée "Test Products" de la base de données.
- Valeurs manquantes : J'ai trouvée apres jointure que la base de données "produits" manquait de valeurs pour le produit "0_2245" mais il y avait les transaction pour ce produit (elles ont été spesifiées left_only) . Je vais stocker ses transactions car elles sont importantes pour l'analyse.
- Changement le type de date de 'objet' en 'date'

```
6]:  # Première jointure outer pour transaction et products
merge= pd.merge(trans, prod, on = 'id_prod', how = 'outer', indicator = True)
# Seconde jointure inner pour customers et le premier DF
merge_2= pd.merge(castom, merge, on= 'client_id', how='inner')
merge_2.head()
```

Out[6]:

	client_id	sex	birth	id_prod	date	session_id	price	categ	_merge
0	c_4410	f	1967	0_1277	2022-03-25 00:03:39.156997	s_184041	7.99	0.0	both
1	c_4410	f	1967	0_1277	2021-09-25 00:03:39.156997	s_94984	7.99	0.0	both
2	c_4410	f	1967	0_1376	2021-09-24 22:58:27.418343	s_94984	16.24	0.0	both
3	c_4410	f	1967	0_1376	2022-04-24 22:58:27.418343	s_198987	16.24	0.0	both
4	c_4410	f	1967	0_1376	2023-01-24 22:58:27.418343	s_331878	16.24	0.0	both

Nettoyage le jeu de données

Produits de test

```
#détection de données "test"
merge_2.loc[merge_2['id_prod']=='T_0',:]
```

```
7]:
```

	client_id	sex	birth	id_prod		date	session_id	price	categ	_merge
218171	ct_0	f	2001	T_0	test_2021-03-01 02:30:02.237419		s_0	-1.0	0.0	both
218172	ct_0	f	2001	T_0	test_2021-03-01 02:30:02.237425		s_0	-1.0	0.0	both
218173	ct_0	f	2001	T_0	test_2021-03-01 02:30:02.237436		s_0	-1.0	0.0	both
218174	ct_0	f	2001	T_0	test_2021-03-01 02:30:02.237430		s_0	-1.0	0.0	both
218175	ct_0	f	2001	T_0	test_2021-03-01 02:30:02.237449		s_0	-1.0	0.0	both

```
# suppression des donnée contenant "test"
merge_2.drop(merge_2[merge_2['date'].str.contains('test')].index,inplace = True)
merge_2.loc[merge_2['id_prod']=='T_0',:]
```

```
8]:
```

	client_id	sex	birth	id_prod	date	session_id	price	categ	_merge
--	-----------	-----	-------	---------	------	------------	-------	-------	--------

Valeurs manquantes

```
prod.loc[prod['id_prod']=='0_2245',:]
```

```
9]:
```

	id_prod	price	categ
--	---------	-------	-------

```
# id_prod 0_2245 dans la base 'transaction'
trans.loc[trans['id_prod']=='0_2245',:]
```

```
0]:
```

	id_prod		date	session_id	client_id
2633	0_2245	2022-09-23 07:22:38.636773	s_272266	c_4746	
10106	0_2245	2022-07-23 09:24:14.133889	s_242482	c_6713	
11727	0_2245	2022-12-03 03:26:35.696673	s_306338	c_5108	
15675	0_2245	2021-08-16 11:33:25.481411	s_76493	c_1391	
16377	0_2245	2022-07-16 05:53:01.627491	s_239078	c_7954	

```
# changement de type de date
merge_2['date'] = pd.to_datetime(merge_2['date'])
merge_2.dtypes
```

```
32]:
```

client_id	object
sex	object
birth	int64
id_prod	object
date	datetime64[ns]

Une analyse des différents indicateurs de vente

Calcul du chiffre d'affaires, différents indicateurs et graphiques autour d'eux (la demande d'Antoine)

- Total du chiffre d'affaires : 11 853 728.68
- Taux d'évolution du chiffre d'affaires : *Grace à cet indice on peut voir la tendance globale pour toute la période d'analyse.*
- *Il y a une diminution. -5.12 %.*

```
[33]: total_ca= merge_2["price"].sum()
      n=len(merge_2)
      print("Total du chiffre d'affaires :", "{:.2f}".format(total_ca))

Total du chiffre d'affaires : 11853728.68
```

```
[35]: oldest=df1[(df1['date'] <= '2022-03')]['price'].sum() # pour premier 12 mois
      newest=df1[(df1['date'] >= '2022-03') & (df1['date'] <= '2023-02')]['price'].sum() # pour deuxieme 12 mois
      print(" ""Chiffre d'affaires de 2021-03 a 2022-02:", "{:.2f}".format(oldest),'\n',"Chiffre d'affaires de 2022-03 a 2023-02:", "{:.2f}".format(newest))
      print("Taux d'évolution:", "{:.2f}".format((newest-oldest)*100/oldest), "%")

Chiffre d'affaires de 2021-03 a 2022-02: 6347193.58
Chiffre d'affaires de 2022-03 a 2023-02: 6021991.63
Taux d'évolution: -5.12 %
```

Une analyse des différents indicateurs de vente

- L'évolution dans le temps et une décomposition en moyenne mobile pour évaluer la tendance globale

Il n'y a pas de fortes fluctuations du chiffre d'affaires lors de la comparaison des moyennes pour différentes périodes

Chiffre d'affaires moyen pour la période différente

```
# CA moyenne par année
result = df1.groupby((df1.date.dt.year - 1) / 2 * 2 + 1).price.mean()
result
```

```
date
2021.0    477082.6560
2022.0    509056.8175
2023.0    487110.1550
Name: price, dtype: float64
```

```
# CA moyenne par période
def incomeFromRange(yr1, yr2):
    return df1[df1.date.dt.year.between(yr1, yr2)].price.mean()
#Then generate the result as:
result = pd.DataFrame([ [f'{yr}-{yr+1}', incomeFromRange(yr, yr+1)]
                        for yr in range(df1.date.dt.year.min(), df1.date.dt.year.max()) ],
                      columns=['date', 'price'])
result
```

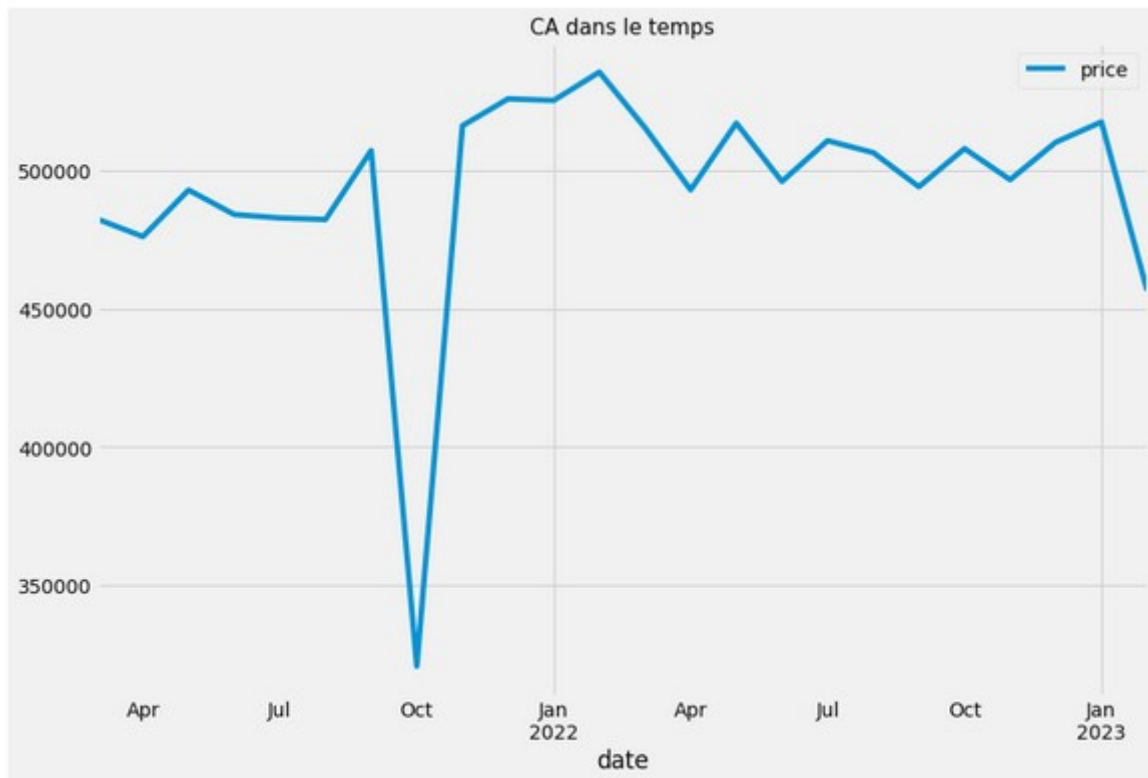
	date	price
0	2021-2022	494523.107727
1	2022-2023	505921.580000

```
# CA moyenne pour les 12 mois premiers et pour les 12 mois suivants
df_selection=df1[(df1['date'] <= '2022-03')]['price'].mean()
df_selection1=df1[(df1['date'] >= '2022-03') & (df1['date'] <= '2023-02')]['price'].mean()
print(" ""Chiffre d'affaires moyen de 2021-03 a 2022-02:", "{:.2f}".format(df_selection),'\n',"Chiffre d'affaires moyen de 2022-03 a 2023-02:", "{:.2f}".format(df_selection1))
```

Chiffre d'affaires moyen de 2021-03 a 2022-02: 488245.66
Chiffre d'affaires moyen de 2022-03 a 2023-02: 501832.64

Une analyse des différents indicateurs de vente

- Evolution dans le temps du Chiffre d'affaires avec graphiques. * *Il y a une fort baisse au mois d'octobre 2021.*



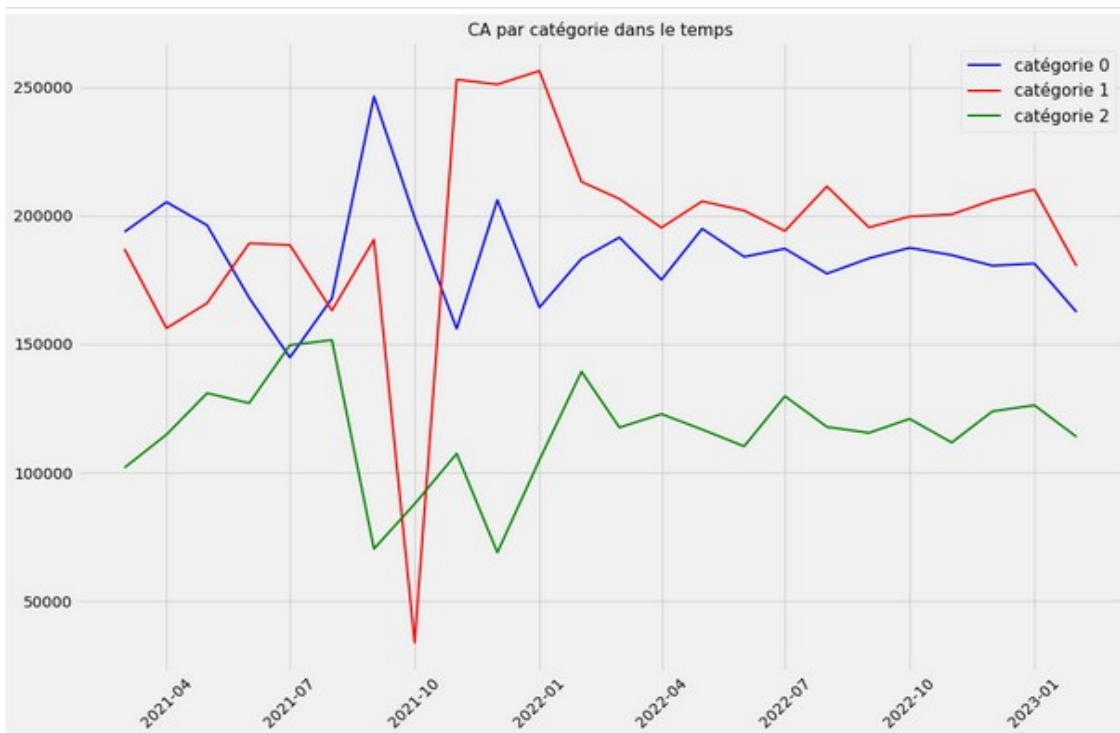
```
import matplotlib
import matplotlib.pyplot as plt
import matplotlib.style as style
%matplotlib inline
style.use('fivethirtyeight')
ca_graph = df1.plot(x = 'date', y = 'price', figsize = (12,8))
plt.title(" CA dans le temps " , fontsize = 15 )
```

```
#echentillon avec regroupement par mois pour constuire graphique
per = merge_2.date.dt.to_period("M")
df1 = merge_2.groupby([per])['price'].sum().reset_index()
df1.head()
```

	date	price
0	2021-03	482440.61
1	2021-04	476109.30
2	2021-05	492943.47
3	2021-06	484088.56
4	2021-07	482835.40

Une analyse des différents indicateurs de vente

- L'évolution du chiffre d'affaires par catégories. *Nous constatons une forte baisse du chiffre d'affaires de la catégorie 1 en octobre 2021. Mais malgré cela il a la plus grande part du chiffre d'affaire parmi les autres catégories.*



```
#echantillon avec regroupement par mois et categorie pour constr.graph
df2 = merge_2.groupby(['categ', 'per'])['price'].sum().reset_index()
df2.head()
```

```
# changement la date en numerique avant la code de graph
df2['date'] = df2['date'].dt.to_timestamp()
```

```
# Chiffre d'affaires par categories
```

```
evol0=df2[(df2["categ"] == 0)]
evol_1=df2[(df2["categ"] == 1)]
evol_2=df2[(df2["categ"] == 2)]
```

```
# Taille de la figure
```

```
plt.figure(figsize=(15,10))
plt.grid(True)
```

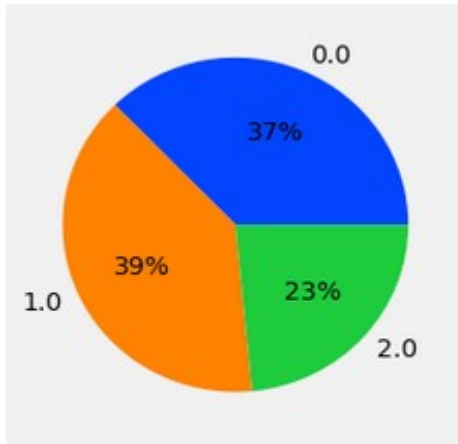
```
plt.plot(evol0['date'], evol0.price, 'b', linewidth = 2, label = 'catégorie 0')
plt.plot(evol_1['date'], evol_1.price, 'r', linewidth = 2, label = 'catégorie 1')
plt.plot(evol_2['date'], evol_2.price, 'g', linewidth = 2, label = 'catégorie 2')
plt.legend(prop = {'size' : 15})
plt.xticks(rotation = 45)
plt.title(" CA par catégorie dans le temps ", fontsize = 15 )
```

```
plt.show()
```


Une analyse des différents indicateurs de vente

- Répartition du chiffre d'affaires par categorie en proportion.

Chiffre d'affaires des categorie 1 et 0 a presque la meme part . Pour la 2ème catégorie, deux fois moins.



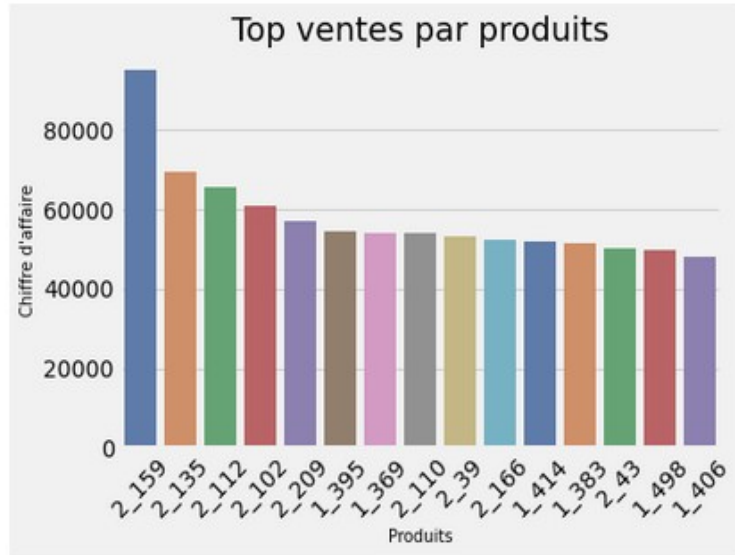
Chiffre d'affaires par categories :

```
categ 2. 2780275.02
categ 0. 4419730.97
categ 1. 4653722.69
```

```
# Pour répondre à ces questions afficher le tableau comme ceci :
evol=merge_2.groupby('categ').sum().reset_index()
#define data
data = evol['price']
labels = evol['categ']
#define Seaborn color palette to use
colors = sns.color_palette('bright')[0:5]
#create pie chart
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.show()
print("Chiffre d'affaires par categories :\n", 'categ 2.',evol_2['price'].sum())
print(' ', 'categ 0.', "{:.2f}".format(evol0['price'].sum()))
print(' ', 'categ 1.', "{:.2f}".format(evol_1['price'].sum()))
```

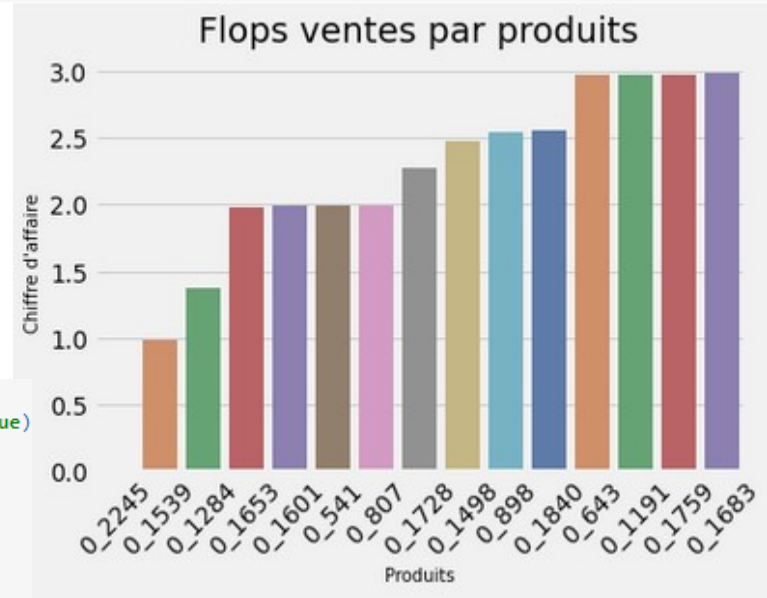
Une analyse des différents indicateurs de vente

- Les Tops et Flops des Ventes par produits :



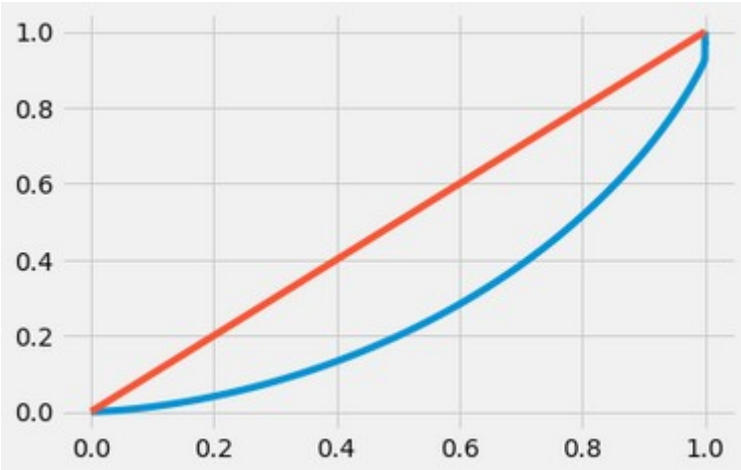
```
# creation DF top ventes pour barplot en ascending True
topVentes = merge_2.groupby(['id_prod']).sum().reset_index().sort_values(by=['price'], ascending=True)
# Utilisation de La Librairie Seaborn head 15
sns.barplot(x=topVentes['id_prod'].head(15), y=topVentes.price, palette='deep')
plt.xlabel('Produits', fontsize=10)
plt.ylabel("Chiffre d'affaire", fontsize=10)
plt.title("Flops ventes par produits", fontsize=20)
plt.xticks(rotation=45)
plt.show
```

```
# creation DF top ventes pour barplot en ascending False
topVentes = merge_2.groupby(['id_prod']).sum().reset_index().sort_values(by=['price'], ascending=False)
# Utilisation de La Librairie Seaborn head 15
sns.barplot(x=topVentes['id_prod'].head(15), y=topVentes.price, palette='deep')
plt.xlabel('Produits', fontsize=10)
plt.ylabel("Chiffre d'affaire", fontsize=10)
plt.title("Top ventes par produits", fontsize=20)
plt.xticks(rotation=45)
plt.show
```



Une analyse des différents indicateurs de vente

- La répartition du chiffre d'affaires entre nos clients, via une courbe de Lorenz. *Il varie entre 0 (égalité parfaite) et 1 (inégalité extrême). Entre 0 et 1, l'inégalité est d'autant plus forte que l'indice de Gini est élevé.*



On peut dire que la concentration n'est pas égalitaire puisque notre courbe n'est pas proche de la première bissectrice et l'indice de Gini est égal 0.446

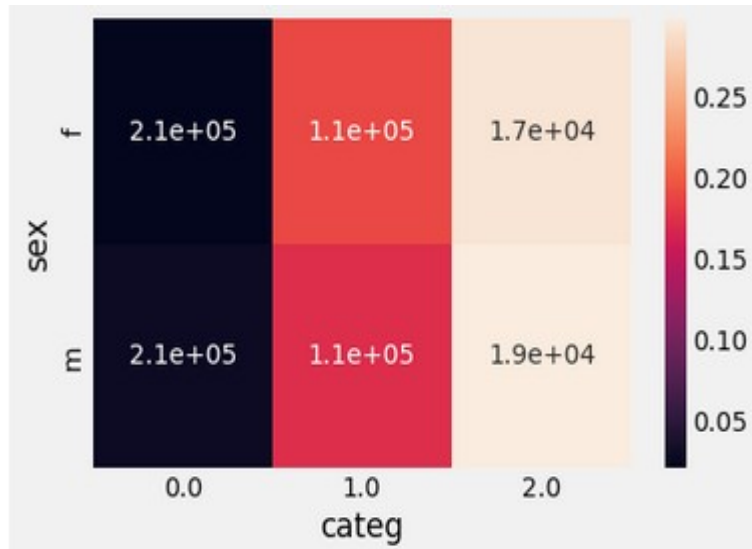
```
# Pour visualiser cela, nous utilisons la courbe de Lorenz.
# plus la courbe de Lorenz est proche de la première bissectrice,
# plus la concentration est égalitaire.
dep = dfCl['price'].values
n = len(dep)
lorenz = np.cumsum(np.sort(dep)) / dep.sum()
lorenz = np.append([0],lorenz) # La courbe de Lorenz commence à 0
xaxis = np.linspace(0-1/n,1+1/n,len(lorenz)) #Il y a un segment de
plt.plot(xaxis,lorenz,drawstyle='steps-post')
plt.plot([0,1], [0,1]) #tracer la bissectrice
plt.show()
```

```
# Le calcul de l'indice de Gini. Le coefficient de GINI permet d'évaluer de façon chiffrée cette répartition.
# Il correspond à deux fois l'aire sous la courbe de Lorenz.
AUC = (lorenz.sum() -lorenz[-1]/2 -lorenz[0]/2)/n # Surface sous la courbe de Lorenz. Le premier segment (lo
S = 0.5 - AUC # surface entre la première bissectrice et le courbe de Lorenz
gini = 2*S
gini
```

0.44638654137401435

Le lien entre le genre des clients et les catégories des livres achetés (la demande de Julie)

- Analyser de deux variables qualitatives. Avec cette heatmap on peut dire que la categorie 0 est la plus achetée , de plus, par les femmes. Et la categorie la moins achetée est 2 , par les hommes.



```
# Le code affichant cette heatmap:
tx = cont.loc[:,["Total"]]
ty = cont.loc[["Total"],:]
n = len(merge_2)
indep = tx.dot(ty) / n

c = cont.fillna(0) # On remplace les valeurs nulles par 0
measure = (c-indep)**2/indep
xi_n = measure.sum().sum()
table = measure/xi_n
sns.heatmap(table.iloc[:,-1],annot=c.iloc[:,-1])
plt.show()
```

```
# Pour répondre à ces questions afficher le tableau comme ceci :
X = "sex"
Y = "categ"
cont = merge_2[[X,Y]].pivot_table(index=X,columns=Y,aggfunc=len,margins=True,margins_name="Total")
cont
```

Le lien entre le genre des clients et les catégories des livres achetés

(Un test statistique pour confirmer la corrélation entre ces variables)

Un test de χ^2 (khi2) dans le cas d'une corrélation entre deux variables qualitatives.

P-Value est égal 1.13 . A un seuil de 5% on rejette L'hypothèse H0 l'indépendance des deux variables . Corelation est confirmer.

```
# Création de la matrice "valeurs observées" (tableau de contingence)
X = "sex"
Y = "categ"
cont2 = merge_2[[X, Y]].pivot_table(index=X, columns=Y, aggfunc=len).fillna(0).copy()
tx = merge_2[X].value_counts()
ty = merge_2[Y].value_counts()
cont2
```

```
#Création de la matrice "valeurs attendues"
tx_df = pd.DataFrame(tx)
tx_df.columns = ["c"]
ty_df = pd.DataFrame(ty)
ty_df.columns = ["c"]
# Valeurs totales observées
n = len(merge_2)
# Produit matriciel. On utilise pd.T pour pivoter une des deux séries.
indep = (tx_df.dot(ty_df.T) / n)
indep
```

```
# Calcul de la matrice "écart au carré normalisé de la valeur attendue VS valeur observée"
# Matrice
freq = (cont2-indep)**2/indep
freq
```

	0.0	1.0	2.0
f	3.519681	26.668027	43.645694
m	3.493582	26.470283	43.322061

```
# Calcul du Chi2. Somme des valeurs de la précédente matrice.
# Cette somme suit une loi du Chi2 à k degrés de liberté.
chi2 = freq.sum().sum()
chi2
```

147.11932715457408

```
#Calcul du khi2 et de la pvalue à partir de la matrice des valeurs observées
#(Partie scipy.stats)
st_chi2, st_p, st_dof, st_exp = st.chi2_contingency(cont2)
```

```
# Chi2
# On retrouve bien la même valeur que calculée manuellement
st_chi2
```

147.11906816131497

```
# Degrés de liberté
# C'est ce nombre de degrés de liberté qui sera utilisé par scipy.stats pour
st_dof
```

<

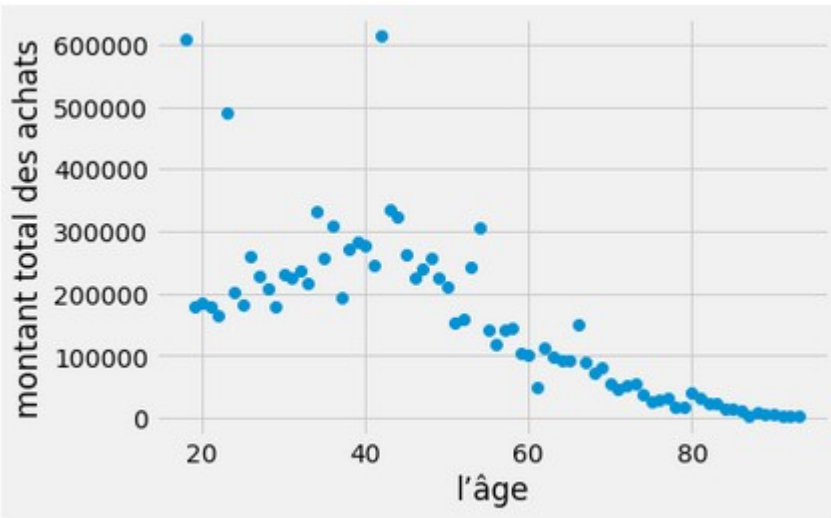
2

```
#Ce qui nous intéresse ici, c'est la variable st_p, qui contient la pvalue.
#Cette valeur nous permet de décider si deux variables sont indépendantes
# ou non en se fixant un seuil de décision.
#L'hypothèse H0 est que les variables sont indépendantes entre elles.
st_p
```

1.1310980597090762e-32

Le lien entre l'âge des clients et le montant total des achats. (la demande de Julie)

Avec l'âge le montant total des achats diminue. Un test statistique de Pearson confirme corrélation négative . Coeficient est $-0,779$.



```
# Calcul l'âge des clients
annéeActuelle=2022
merge_2['age'] = annéeActuelle-merge_2['birth']
merge_2.head()
```

	client_id	sex	birth	id_prod	date	session_id	price	categ	_merge	age
0	c_4410	f	1967	0_1277	2022-03-25 00:03:39.156997	s_184041	7.99	0.0	both	55
1	c_4410	f	1967	0_1277	2021-09-25 00:03:39.156997	s_94984	7.99	0.0	both	55

```
# regroupement par l'age pour calcule Le montant total des achats
df3 = merge_2.groupby(['age'])['price'].sum().reset_index()
```

```
# graphique du Lien entre l'âge des clients et Le montant total des achats
plt.plot(df3['age'],df3['price'],'o')
plt.xlabel("l'âge")
plt.ylabel("montant total des achats")
plt.show()
```

un test de Pearson

- $r = 0$ signifie aucune corrélation

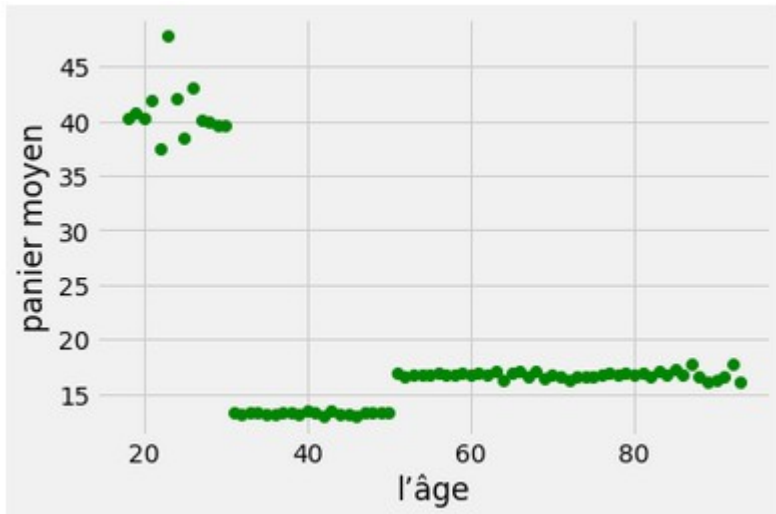
```
from scipy.stats import pearsonr
list1 = df3['age']
list2 = df3['price']
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
```

Pearsons correlation: -0.779

Le lien entre l'âge des clients et la taille du panier moyen

(la demande de Julie)

- Avec l'âge la taille du panier moyen diminue. Un test statistique de Pearson confirme corrélation négative -0,548



```
# # regroupement par l'âge pour calculer la taille du panier moyen  
df4 = merge_2.groupby(['age'],)['price'].mean().reset_index()
```

```
# graphique du lien entre l'âge des clients et la taille du panier moyen  
plt.plot(df4['age'], df4['price'], 'o', color='g')  
plt.xlabel("l'âge")  
plt.ylabel("panier moyen")  
plt.show()
```

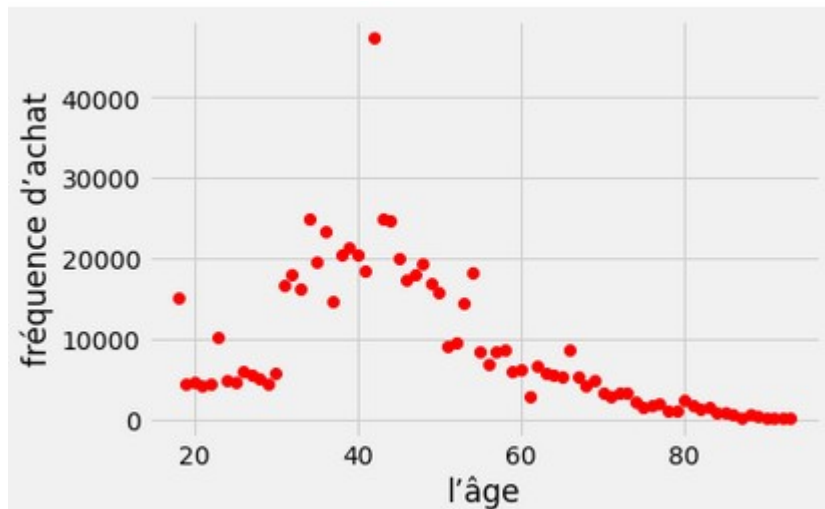
Selon Pearson on confirme corrélation négative.

```
corr, _ = pearsonr(data3, data4)  
print('Pearsons correlation: %.3f' % corr)
```

Pearsons correlation: -0.548

Le lien entre l'âge des clients et la fréquence d'achat (la demande de Julie)

- On confirme corrélation négative. Avec l'âge la fréquence d'achat diminue. Un test statistique de Pearson confirme corrélation négative -0,534



```
# Graphiques qui affiche de Lien entre l'âge des clients et
plt.plot(tab['age'],tab['n'],'o',color='r')
plt.xlabel("l'âge")
plt.ylabel("fréquence d'achat")
plt.show()
```

```
# calcule de la fréquence d'achat par l'age des clients
effectifs = merge_2["age"].value_counts()
modalites = effectifs.index #index de effectifs contient les modalités
# création du tableau à partir des modalités
tab = pd.DataFrame(modalites, columns = ["age"])
tab["n"] = effectifs.values
tab.head()
```

	age	n
0	42	47413
1	34	25005
2	43	24893
3	44	24677
4	36	23475

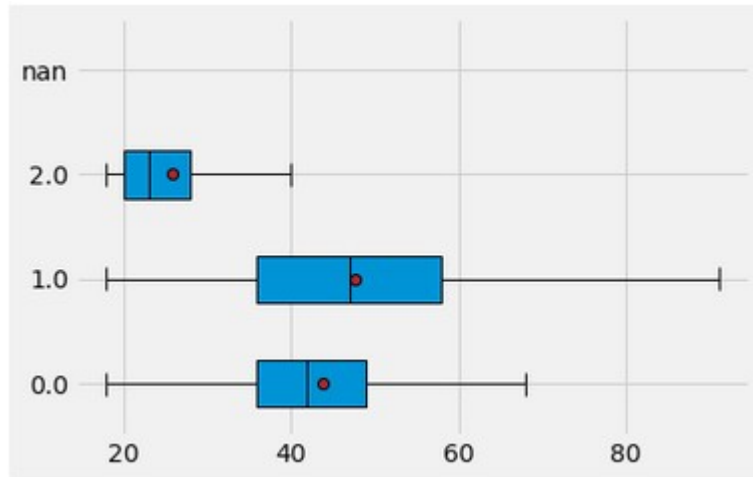
un test de Pearson

```
list3 = tab['age']
list4 = tab['n']
corr, _ = pearsonr(list3, list4)
print('Pearsons correlation: %.3f' % corr)
```

Pearsons correlation: -0.534

Une distribution entre l'âge des clients et les catégories (la demande de Julie)

- Test de comparaison. On rejette l'indépendance des deux variables. La corrélation est confirmée. Les catégories achetées dépendent de l'âge des clients. Les jeunes achètent des livres de catégorie 2.



```
categAge2 = merge_2[(merge_2["categ"]== 2)]["age"]  
categAge1 = merge_2[(merge_2["categ"] == 1)]["age"]
```

```
# On teste tout d'abord l'égalité des variances :  
st.bartlett(categAge2,categAge1)
```

```
BartlettResult(statistic=10586.296663410541, pvalue=0.0)
```

```
# Analysez une variable quantitative et une qualitative  
X = "categ" # qualitative  
Y = "age" # quantitative  
  
modalites = merge_2[X].unique()  
groupes = []  
for m in modalites:  
    groupes.append(merge_2[merge_2[X]==m][Y])  
  
# Propriétés graphiques (pas très importantes)  
medianprops = {'color':"black"}  
meanprops = {'marker':'o', 'markeredgecolor':'black',  
             'markerfacecolor':'firebrick'}  
plt.boxplot(groupes, labels=modalites, showfliers=False, medianprops=medianprops,  
            vert=False, patch_artist=True, showmeans=True, meanprops=meanprops)  
plt.show()
```

On obtient une p-valeur égale 0.0. On constate donc que l'hypothèse d'égalité des moyennes de catégories est rejetée à un niveau de test de 5%.

```
# On teste ensuite l'égalité des moyennes :  
st.ttest_ind(categAge2,categAge1, equal_var=True)
```

```
Ttest_indResult(statistic=-259.3213545920276, pvalue=0.0)
```