Analysez les ventes de votre entreprise

Analyze company's sales

Nettoyage des données

```
# vérifier les valeurs aberrantes et la distribution
# check for outliers and distribution

print(customers['client_id'].describe())
print()
print(customers['sex'].describe())
print()
print(customers['birth'].describe())
```

```
count       8623
unique      8623
top       c_5316
freq           1
Name: client_id, dtype: object

count     8623
unique       2
top          f
freq      4491
Name: sex, dtype: object

count    8623.000000
mean     1978.280877
std        16.919535
min      1929.000000
25%      1966.000000
50%      1979.000000
75%      1992.000000
max      2004.000000
Name: birth, dtype: float64
```

Nous pouvons voir que le cadre de données des clients n'a pas de valeurs manquantes. Il y a plus de clients de sexe féminin que de sexe masculin. Le client le plus âgé est né en 1929, le plus jeune en 2004. Nous pourrions avoir besoin de convertir l'année de naissance en âge pour de meilleurs calculs.

We can see that customers dataframe has no missing values. There are more female customers than male ones. Oldest customer born in 1929, youngest in 2004. We might need to convert Birth Year into Age for better calculations.

Nettoyage des données

```
# vérifier les valeurs aberrantes et la distribution
# check for outliers and distribution

print(products['id_prod'].describe())
print()
print(products['price'].describe())
print()
print(products['categ'].describe())
```

```
count     3287
unique    3287
top       1_14
freq         1
Name: id_prod, dtype: object

count    3287.000000
mean       21.856641
std        29.847908
min        -1.000000
25%         6.990000
50%        13.060000
75%        22.990000
max       300.000000
Name: price, dtype: float64

count    3287.000000
mean        0.370246
std         0.615387
min         0.000000
25%         0.000000
50%         0.000000
75%         1.000000
max         2.000000
Name: categ, dtype: float64
```

Nous pouvons voir que le cadre de données des produits n'a pas non plus de valeurs manquantes. La description montre une anomalie dans la valeur du prix minimal, ce qui doit être examiné.

We can see that products dataframe also has no missing values. Describe shows anomaly in minimal price value, this is needs to be investigated.

Nettoyage des données

```
# vérifier les valeurs aberrantes et la distribution
# check for outliers and distribution

print(transactions['id_prod'].describe())
print()
print(transactions['date'].describe())
print()
print(transactions['session_id'].describe())
print()
print(transactions['client_id'].describe())
```

```
count     337016
unique      3266
top        1_369
freq        1081
Name: id_prod, dtype: object

count                             337016
unique                            336855
top       test_2021-03-01 02:30:02.237413
freq                                  13
Name: date, dtype: object

count     337016
unique    169195
top          s_0
freq         200
Name: session_id, dtype: object

count     337016
unique      8602
top       c_1609
freq       12855
Name: client_id, dtype: object
```

Nous pouvons voir que les transactions dataframe n'ont pas non plus de valeurs manquantes. Cependant, il y a deux choses importantes ici :
- la colonne datetime contient des données "test". Cela doit être examiné.
- Le nombre unique de id_prod ne correspond pas à la base de données des produits (3266 produits dans les transactions, 3287 produits dans la base de données des produits). Cela doit être vérifié.

We can see that transactions dataframe also has no missing values. However 2 important things here are:
- datetime column contains "test" data. This needs to be investigated.
- unique count of id_prod does not match products database (3266 products in transactions, 3287 products in products database). This needs to be investigated.

Nettoyage des données

```
# trouver les données de test dans la colonne date
# find test data in date col

transactions[transactions['date'].apply(lambda date: 'test' in date)]
```

	id_prod	date	session_id	client_id
1431	T_0	test_2021-03-01 02:30:02.237420	s_0	ct_1
2365	T_0	test_2021-03-01 02:30:02.237446	s_0	ct_1
2895	T_0	test_2021-03-01 02:30:02.237414	s_0	ct_1
5955	T_0	test_2021-03-01 02:30:02.237441	s_0	ct_0
7283	T_0	test_2021-03-01 02:30:02.237434	s_0	ct_1
...	...	...	...	...
332594	T_0	test_2021-03-01 02:30:02.237445	s_0	ct_0
332705	T_0	test_2021-03-01 02:30:02.237423	s_0	ct_1
332730	T_0	test_2021-03-01 02:30:02.237421	s_0	ct_1
333442	T_0	test_2021-03-01 02:30:02.237431	s_0	ct_1
335279	T_0	test_2021-03-01 02:30:02.237430	s_0	ct_0

200 rows × 4 columns

```
# filtrer les lignes de test
# filter out test rows

transactions = transactions[~transactions['date'].apply(lambda date: 'test' in date)]
```

Nettoyage des données

```
# make a list of unique product ids, print out total number of unique ids
# faire une liste d'identifiants uniques de produits, imprimer le nombre total d'identifiants uniques

prod_list = list(products['id_prod'].unique())
print('Products database has ' + str(products['id_prod'].nunique()) + ' unique items.')

purchased_prod_list = list(transactions['id_prod'].unique())
print('Transactions database has ' + str(transactions['id_prod'].nunique()) + ' unique items.')
```

```
Products database has 3286 unique items.
Transactions database has 3265 unique items.
```

```
# les articles qui ont été achetés mais qui n'existent pas dans la base de données des produits
# items that were bought but dont exist in product database

lost_item = list(set(purchased_prod_list).difference(prod_list))
print(lost_item)
```

```
['0_2245']
```

```
# Des articles qui n'ont jamais été achetés
# items that were never bought

never_purchased = list(set(prod_list).difference(purchased_prod_list))
print(never_purchased)
```

```
['0_1800', '1_0', '0_1318', '0_1780', '0_310', '0_1062', '0_299', '2_87', '0_1025', '0_510', '0_322', '0_525', '0_1119', '2_86', '0_2308', '0_1645', '0_1014', '0_1620', '0_1624', '1_3
94', '2_72', '0_1016']
```

```
# vérifier si ces produits existent dans nos données brutes
# check if those products exist in our raw data

print(lost_item[0] in transactions['id_prod'].values)
print(lost_item[0] in products['id_prod'].values)
print(never_purchased[0] in products['id_prod'].values)
```

```
True
False
True
```

L’analyse des données

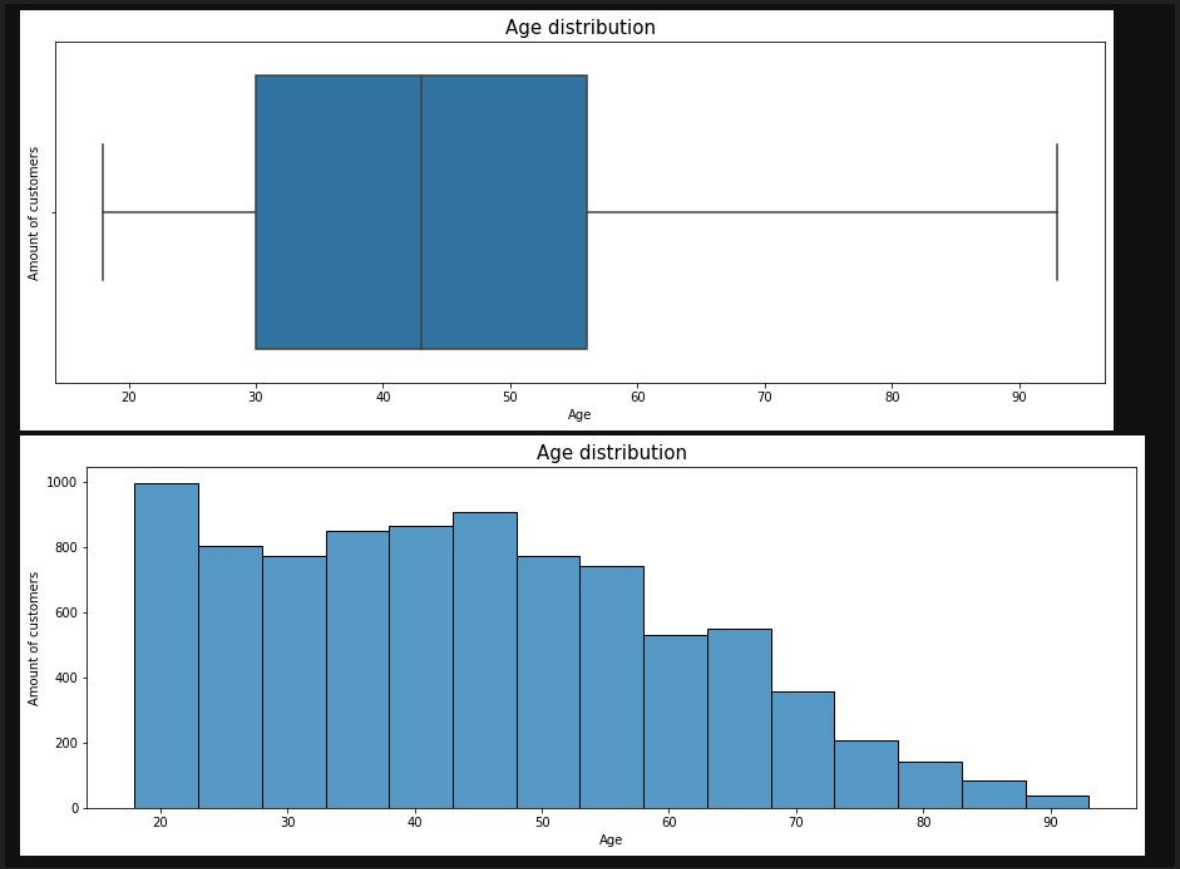
```
# comprendre nos clients, âge moyen et âge le plus fréquent
# understanding our customers, average age and most frequent age

print('Average age of registered user is ' + str(round(customers['age'].mean(), 1)))
print('Most frequent age among registered users is ' + str(round(customers['age'].mode()[0], 1)))
print('\n')
print('Average age of our customers is ' + str(round(df_global['age'].mean(), 1)))
print('Most frequent age of our customers is ' + str(round(df_global['age'].mode()[0], 1)))
```

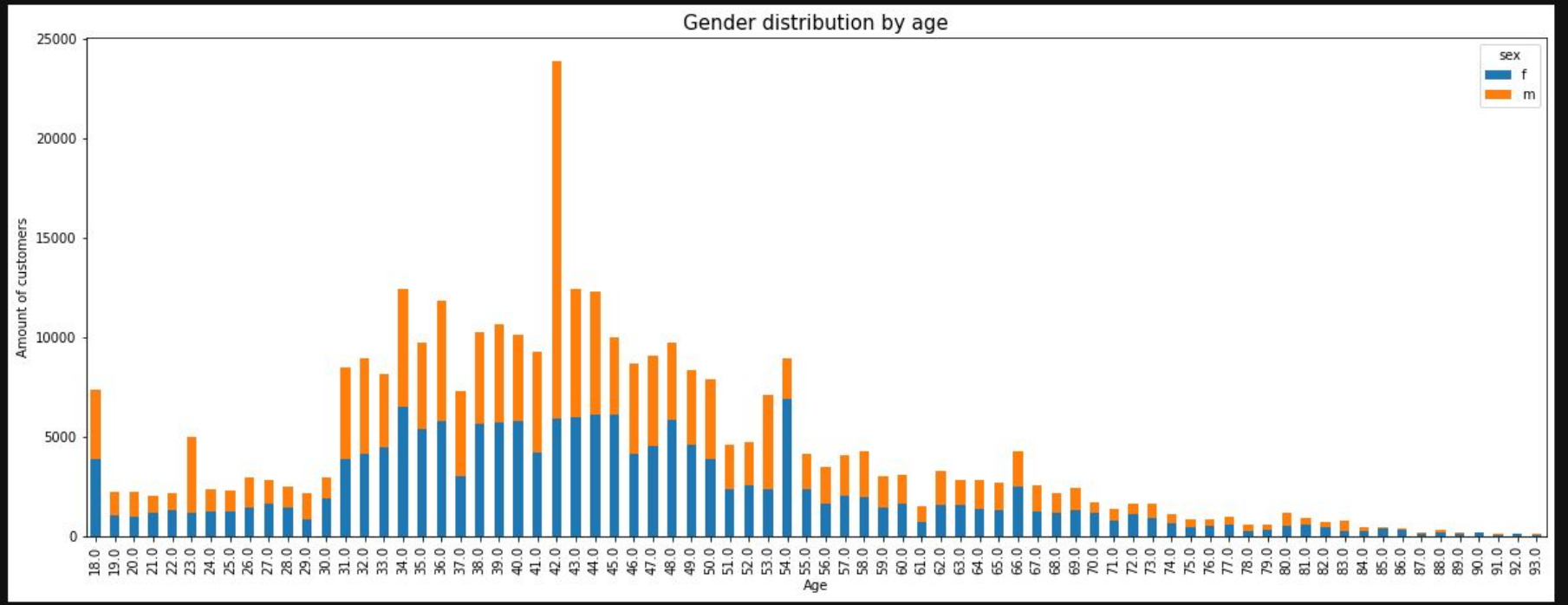
```
Average age of registered user is 43.7
Most frequent age among registered users is 18


Average age of our customers is 44.2
Most frequent age of our customers is 42.0
```

L’analyse des données



L’analyse des données



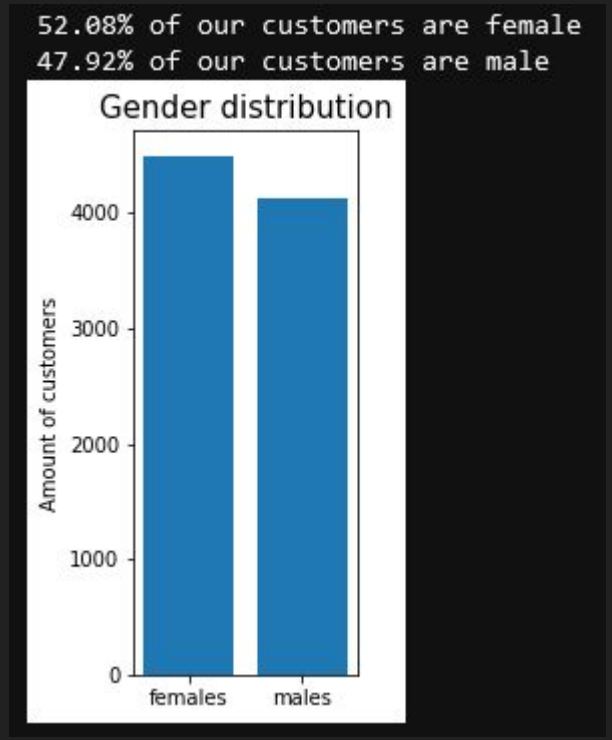
L’analyse des données

Conclusions 1 (l'age):

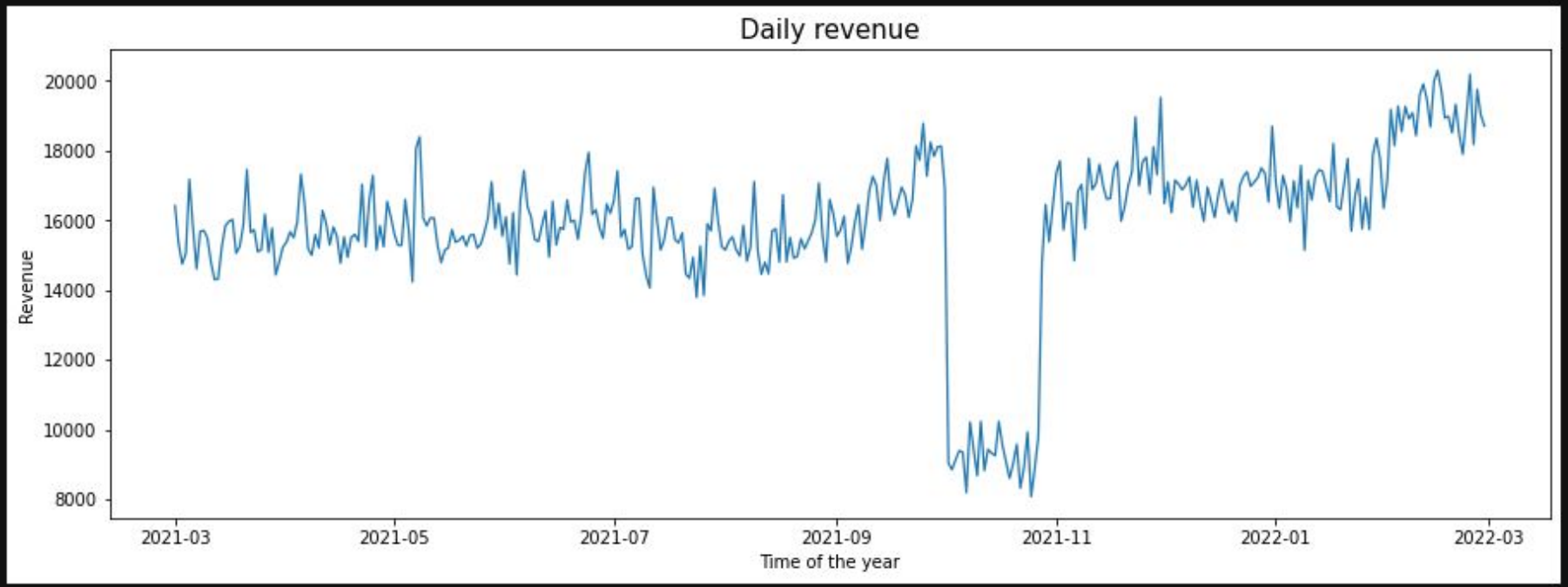
L'âge moyen de nos clients est de 44 ans. L'âge le plus populaire parmi nos clients est 42 ans (toutefois, parmi les utilisateurs enregistrés, l'âge le plus populaire est 18 ans).

La visualisation nous montre que la majorité des clients ont entre 31 et 54 ans. Notre plus grand groupe de clients est constitué de personnes âgées de 18 à 23 ans.

La répartition par sexe est légèrement en faveur des femmes (52%), par rapport aux hommes (48%).
Si nous comparons la répartition par âge et par sexe, la tendance à l'égalité se maintient. Toutefois, quelques tranches d'âge sont dominées par un sexe spécifique (par exemple, les femmes de 54 ans, les hommes de 42 ans, les hommes de 23 ans).

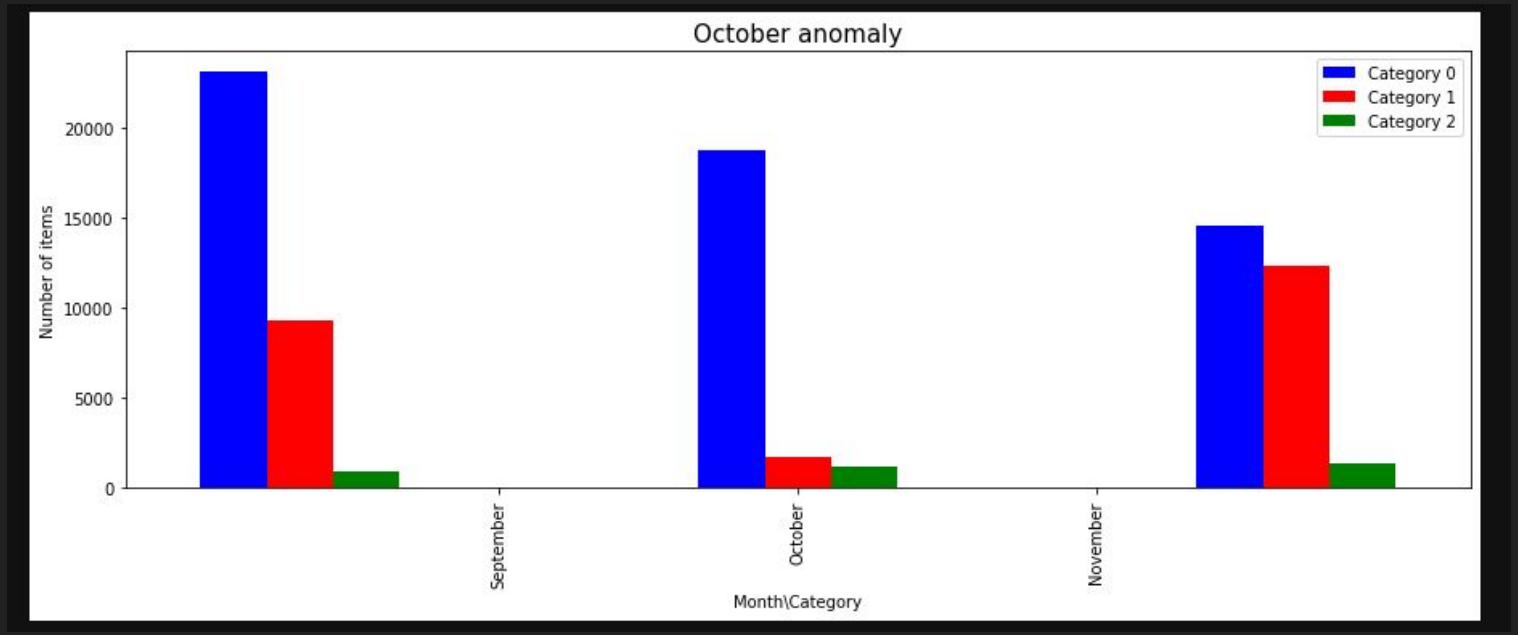


L’analyse des données



Ce graphique représente nos revenus sur l'année. Nous pouvons voir que la tendance générale est à l'augmentation. Mais il y a une nette baisse en octobre, avec une reprise immédiate en novembre. Nous devons rechercher les causes probables et voir si cela se reproduira en octobre prochain.

L’analyse des données



L’analyse des données

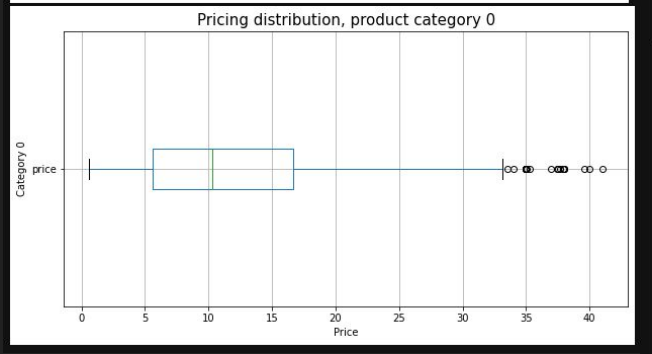
Conclusions 1 (sur le mois d'octobre):

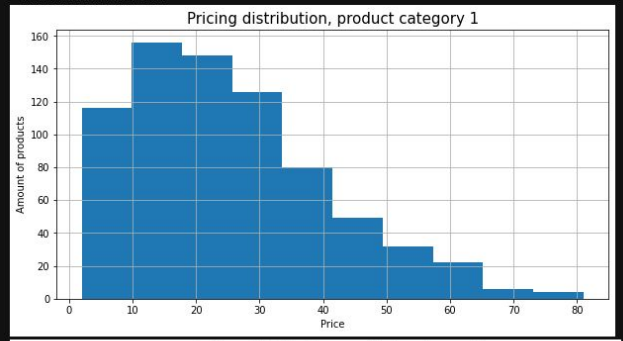
La comparaison indique que l'anomalie d'octobre pourrait être liée aux ventes d'articles de la catégorie 1. Les données montrent un certain nombre de jours où cette catégorie n'a pas été vendue du tout. Les raisons potentielles de ce phénomène sont:

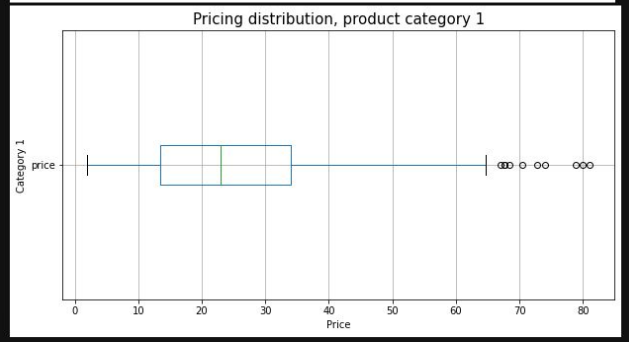
- corruption des données (les données sont incorrectes, la catégorie 1 a été vendue mais n'a jamais été enregistrée dans la base de données)

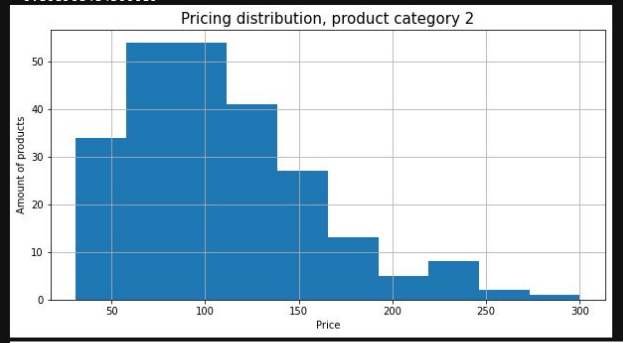
- anomalie des ventes (les données sont correctes, les clients n'ont pas acheté la catégorie 1 en octobre autant qu'avant ou après).

L’analyse des données









L’analyse des données

Conclusions 2 (la tarification):

Le prix moyen de nos produits est de 21,9 EUR. Le prix le plus courant est de 4,99 EUR.

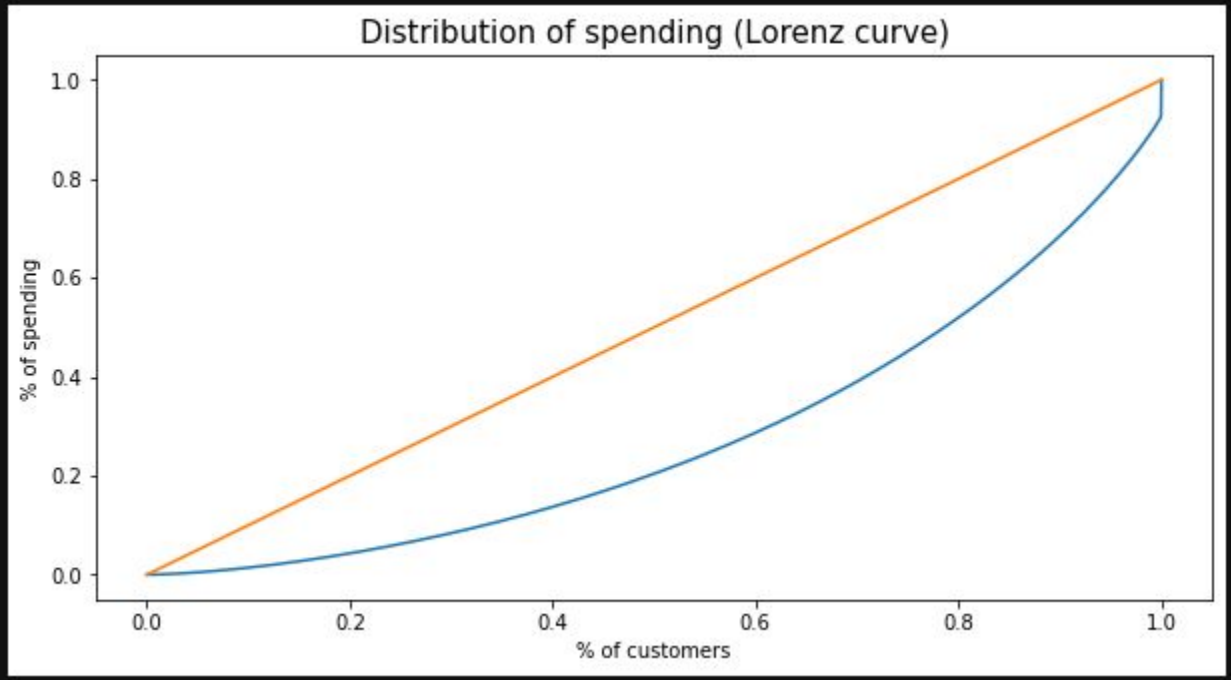
L'analyse de chaque catégorie de produits:
La catégorie de produits la moins chère est "0" avec un prix moyen de 11,7 EUR. La majorité des produits de cette catégorie ont un prix compris entre 5,6 et 16,7 EUR. Le prix le plus courant est de 4,99 EUR.

La catégorie "1" est la deuxième plus abordable, avec un prix moyen de 25,5 EUR. Le prix le plus courant est de 22,99 EUR. La majorité des produits de cette catégorie ont un prix compris entre 13,4 et 33,99 EUR.

La catégorie "2" regroupe nos produits haut de gamme. Le prix moyen est de 108 EUR. Le prix le plus courant est de 50,99 EUR. La majorité des produits de cette catégorie ont un prix compris entre 71 et 137 EUR.

L’analyse des données

```
Gini index is 0.4397099070275077
```



L’analyse des données

Conclusions 3 (répartition des richesses):

Nous pouvons voir ici la liste des clients qui ont dépensé le plus au cours de cette année. Il y a clairement 4 clients qui dépensent beaucoup plus que les autres, nous devons les considérer comme des valeurs aberrantes pour l'analyse.

En utilisant l'indice de Gini et la courbe de Lorenz, nous pouvons voir la distribution des dépenses parmi nos clients. La visualisation nous montre une distribution normale (ligne bleue) qui n'est pas très éloignée de l'idéal (ligne orange). L'indice de Gini de 0,43 confirme que l'inégalité est moyenne (un indice de Gini de 0 correspond à une distribution parfaite, 1 à une inégalité maximale).