ANALYSE DU STOCK ET DES VENTES DU SITE BOTTLENECK

OBJECTIF DE CE NOTEBOOK

Bienvenue dans l'outil plébiscité par les analystes de données Jupyter.

Il s'agit d'un outil permettant de mixer et d'alterner codes, textes et graphique.

Cet outil est formidable pour plusieurs raisons:

- il permet de tester des lignes de codes au fur et à mesure de votre rédaction, de constater immédiatement le résultat d'un instruction, de la corriger si nécessaire.
- De rédiger du texte pour expliquer l'approche suivie ou les résultats d'une analyse et de le mettre en forme grâce à du code html ou plus simple avec **Markdown**
- d'agrémenter de graphiques

Pour vous aider dans vos premiers pas à l'usage de Jupyter et de Python, nous avons rédigé ce notebook en vous indiquant les instructions à suivre.

Il vous suffit pour cela de saisir le code Python répondant à l'instruction donnée.

Vous verrez de temps à autre le code Python répondant à une instruction donnée mais cela est fait pour vous aider à comprendre la nature du travail qui vous est demandée.

Et garder à l'esprit, qu'il n'y a pas de solution unique pour résoudre un problème et qu'il y a autant de résolutions de problèmes que de développeurs ;)...

Etape 1 - Importation des librairies et chargement des fichiers

1.1 - Importation des librairies

Entrée [1]:
```
#Importation de la librairie Pandas
import pandas as pd
import numpy as np
```

Entrée [2]:
```
#Importation de la librairie plotly express
import plotly.express as px
```

Entrée [3]:
```
#Trouver dans Google l'instruction permettant d'afficher toutes les colonnes d'un dataframe
#Saisir, dans Google, les mots clés "display all columns dataframe Pandas", par exemple.
#Dans les résultats de la recherche, privilégiez les solutions provenants de Stack Overflow ou Medium
pd.set_option('display.max_columns', None)
```

1.2 - Chargements des fichiers

Entrée [4]:
```
#Importation du fichier web.xlsx
df_web = pd.read_excel("web.xlsx")
#Importation du fichier erp.xlsx
df_erp = pd.read_excel("erp.xlsx")
#importation du fichier liaison.xlsx
df_liaison = pd.read_excel("liaison.xlsx")
```

```
C:\Users\maxen\anaconda3\Lib\site-packages\openpyxl\worksheet\_read_only.py:79: UserWarning: Unknown extension is not supported and wil
l be removed
  for idx, row in parser.parse():
C:\Users\maxen\anaconda3\Lib\site-packages\openpyxl\worksheet\_read_only.py:79: UserWarning: Unknown extension is not supported and wil
l be removed
  for idx, row in parser.parse():
C:\Users\maxen\anaconda3\Lib\site-packages\openpyxl\worksheet\_read_only.py:79: UserWarning: Unknown extension is not supported and wil
l be removed
  for idx, row in parser.parse():
```

Etape 2 - Analyse exploratoire des fichiers

2.1 - Analyse exploratoire du fichier erp.xlsx

Entrée [5]:
```
#Afficher les dimensions du dataset
print("les dimmensions du dataset erp sont :", df_erp.shape)
```

```
les dimmensions du dataset erp sont : (825, 6)
```

Entrée [6]:
```
#Consulter le nombre de colonnes
print("Le tableau erp comporte {} colonne(s)".format(df_erp.shape[1]))
#La nature des données dans chacune des colonnes (ndd)
ndd = df_erp.dtypes
print(ndd)
#Le nombre de valeurs présentes dans chacune des colonnes (ndv)
ndv = df_erp.count()
print(ndv)
```

```
Le tableau erp comporte 6 colonne(s)
product_id          int64
onsale_web          int64
price             float64
stock_quantity      int64
stock_status       object
purchase_price    float64
dtype: object
product_id        825
onsale_web        825
price             825
stock_quantity    825
stock_status      825
purchase_price    825
dtype: int64
```

Entrée [7]:
```
#Afficher les 5 premières lignes de la table
df_erp.head()
```

Out[7]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price
0	3847	1	24.2	16	instock	12.88
1	3849	1	34.3	10	instock	17.54
2	3850	1	20.8	0	outofstock	10.64
3	4032	1	14.1	26	instock	6.92
4	4039	1	46.0	3	outofstock	23.77

Entrée [8]:
```
#Vérifier si il y a les lignes en doublons dans la colonne product_id
doublon = df_erp[df_erp['product_id'].duplicated(keep=False)]
nb_doublon = doublon.shape[0]
print("Nombre de doublons :",nb_doublon)
```

```
Nombre de doublons : 0
```

Entrée [9]:
```
#Afficher les valeurs distinctes de la colonne stock_status
#À quelle(s) autre(s) colonne(s) sont-elles liées ?
val_dist = df_erp['stock_status'].unique()
print("Les valeurs distintctes sont :",val_dist)
print("Elles sont liées à la colonne 'stock_quantity'")
```

```
Les valeurs distintctes sont : ['instock' 'outofstock']
Elles sont liées à la colonne 'stock_quantity'
```

Entrée [10]:
```
#Création d'une colonne "stock_status_2
#La valeur de cette deuxième colonne sera fonction de la valeur dans la colonne "stock_quantity"
#si la valeur de la colonne "stock_quantity" est nulle renseigner "outofstock" sinon mettre "instock"
df_erp['stock_status_2'] = np.where(df_erp['stock_quantity'] == 0, 'outofstock', 'instock')
df_erp
```

Out[10]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
0	3847	1	24.2	16	instock	12.88	instock
1	3849	1	34.3	10	instock	17.54	instock
2	3850	1	20.8	0	outofstock	10.64	outofstock
3	4032	1	14.1	26	instock	6.92	instock
4	4039	1	46.0	3	outofstock	23.77	instock
...	...	...	...	...	...	...	...
820	7203	0	45.0	30	instock	23.48	instock
821	7204	0	45.0	9	instock	24.18	instock
822	7247	1	54.8	6	instock	27.18	instock
823	7329	0	26.5	14	instock	13.42	instock
824	7338	1	16.3	40	instock	8.00	instock

825 rows × 7 columns

Entrée [11]:
```
#Vérifions que les 2 colonnes sont identiques:
#Les 2 colonnes sont strictement identiques si les valeurs de chaque ligne sont strictement identiques 2 à 2
#La comparaison de 2 colonnes peut se réaliser simplement avec l'instruction ci-dessous:
df_comp = df_erp["stock_status"] == df_erp["stock_status_2"]
df_comp
#Le résultat est l'affichage de True ou False pour chacune des lignes du dataset
#C'est un bon début, mais difficile à exploiter
```

Out[11]:
```
0       True
1       True
2       True
3       True
4      False
       ...
820     True
821     True
822     True
823     True
824     True
Length: 825, dtype: bool
```

Entrée [12]:
```
#Mais il est possible de synthétiser ce résultat en effectuant la somme de cette colonne:
#True vaut 1 et False 0
#Nous devrions obtenir la somme de 824 qui correspond au nombre de lignes dans ce dataset
df_comp = df_comp.replace({True : 1, False :0}).sum()
df_comp
```

Out[12]:
```
821
```

Entrée [13]:
```
#Si les colonnes ne sont absolument pas identiques ligne à ligne alors identifier la ligne en écart
##Dans ce cas je vous ce lien pour apprendre à réaliser des filtres dans Pandas:
##https://bitbucket.org/hrojas/learn-pandas/src/master/
##Lesson 3
diff = df_erp[df_erp['stock_status'] != df_erp['stock_status_2']]
diff
```

Out[13]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
4	4039	1	46.0	3	outofstock	23.77	instock
398	4885	1	18.7	0	instock	9.66	outofstock
449	4973	0	10.0	-10	outofstock	4.96	instock
573	5700	1	44.5	-1	outofstock	22.30	instock

Entrée [14]:
```
#Corriger la ou les données incohérentes
df_erp['stock_status'] = np.where(df_erp['stock_quantity'] <= 0, 'outofstock', 'instock')
df_erp
#Verification en utilisant le même code que plus haut pour afficher les problemes
```

Out[14]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
0	3847	1	24.2	16	instock	12.88	instock
1	3849	1	34.3	10	instock	17.54	instock
2	3850	1	20.8	0	outofstock	10.64	outofstock
3	4032	1	14.1	26	instock	6.92	instock
4	4039	1	46.0	3	instock	23.77	instock
...	...	...	...	...	...	...	...
820	7203	0	45.0	30	instock	23.48	instock
821	7204	0	45.0	9	instock	24.18	instock
822	7247	1	54.8	6	instock	27.18	instock
823	7329	0	26.5	14	instock	13.42	instock
824	7338	1	16.3	40	instock	8.00	instock

825 rows × 7 columns

Entrée [15]:
```
df_comp2 = df_erp["stock_status"] == df_erp["stock_status_2"]
df_comp2 = df_comp2.replace({True : 1, False :0}).sum()
df_comp2
```

Out[15]: 823

Entrée [16]:
```
diff = df_erp[df_erp['stock_status'] != df_erp['stock_status_2']]
diff
```

Out[16]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
449	4973	0	10.0	-10	outofstock	4.96	instock
573	5700	1	44.5	-1	outofstock	22.30	instock

Entrée [17]:
```
#Vérif
df_erp.loc[df_erp['product_id'] == 4973]
```

Out[17]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
449	4973	0	10.0	-10	outofstock	4.96	instock

2.1.1 - Analyse exploratoire de chaque variable du fichier erp.xlsx

2.1.1.1 - Analyse de la variable PRIX

Entrée [18]:
```
################
## LES PRIX  ##
################
```

Entrée [19]:
```
#Vérification des prix: Y a t-il des prix non renseignés, négatif ou nul?
prob_nul = df_erp[df_erp['price'].isnull()]
prob_nul
```

Out[19]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2

Entrée [20]:
```
#Vérification des prix: Y a t-il des prix non renseignés, négatif ou nul?
prob_0 = df_erp[df_erp['price'] <= 0]
prob_0
```

Out[20]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
151	4233	0	-20.0	0	outofstock	10.33	outofstock
469	5017	0	-8.0	0	outofstock	4.34	outofstock
739	6594	0	-9.1	19	instock	4.61	instock

Entrée [21]:
```
#Afficher le ou les prix non renseignés dans la colonne "price"
prob = df_erp['price'].isnull().sum()
print("Nombres d'article avec un prix non renseignés: {}".format(prob)) #Saisir l'instruction manquante dans la fonction format
```

```
Nombres d'article avec un prix non renseignés: 0
```

Entrée [22]:
```
#Afficher le prix minimum de la colonne "price"
prix_min = df_erp['price'].min()
print('Le prix minimun de la colonne price est de:',prix_min,'€')
```

```
Le prix minimun de la colonne price est de: -20.0 €
```

Entrée [23]:
```
#Afficher le prix maximum de la colonne "price"
prix_max = df_erp['price'].max()
print('Le prix minimun de la colonne price est de:',prix_max,'€')
```

```
Le prix minimun de la colonne price est de: 225.0 €
```

Entrée [97]:
```
#Affichier les prix inférieurs à 0 (qu'est ce qu'il faut en faire ?)
prob_inf = df_erp[df_erp['price'] <0]
prob_inf
print("mettre les prix à 0 si le prix est inférieur à 0")
print("Vérifier le produit 6594 qui à un prix à -9.1 et un stock à 19. Je vais le mettre en rupture, me donner les info sur le bon prix")
```

```
mettre les prix à 0 si le prix est inférieur à 0
Vérifier le produit 6594 qui à un prix à -9.1 et un stock à 19. Je vais le mettre en rupture, me donner les info sur le bon prix
```

Entrée [98]:
```
df_erp.loc[df_erp['price']<0, 'price']=0
df_erp.loc[df_erp['product_id'] == 6594, 'stock_status'] = 'outofstock'
```

Entrée [99]:
```
df_erp.loc[df_erp['product_id'] == 6594]
```

Out[99]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price
739	6594	0	0.0	19	outofstock	4.61

2.1.1.2 - Analyse de la variable STOCK

Entrée [27]:
```
#######################
### stock_quantity  ###
#######################
#Vérification de la colonne stock quantity
```

Entrée [28]:
```
#Afficher la quantité minimum de la colonne "stock_quantity"
stock_min = df_erp['stock_quantity'].min()
stock_min
```

Out[28]: -10

Entrée [29]:
```
#Afficher la quantité maximum de la colonne "stock_quantity"
stock_max = df_erp['stock_quantity'].max()
stock_max
```

Out[29]: 145

Entrée [30]:
```
#Affichier les stocks inférieurs à 0 (qu'est ce qu'il faut en faire ?)
stock_inf = df_erp[df_erp['stock_quantity'] < 0]
stock_inf
#Mettre les stock à 0
```

Out[30]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
449	4973	0	10.0	-10	outofstock	4.96	instock
573	5700	1	44.5	-1	outofstock	22.30	instock

Entrée [31]:
```
#Suppression des stock negatif
df_erp['stock_quantity'] = df_erp['stock_quantity'].where(df_erp['stock_quantity'] >0,0)
df_erp
```

Out[31]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
0	3847	1	24.2	16	instock	12.88	instock
1	3849	1	34.3	10	instock	17.54	instock
2	3850	1	20.8	0	outofstock	10.64	outofstock
3	4032	1	14.1	26	instock	6.92	instock
4	4039	1	46.0	3	instock	23.77	instock
...	...	...	...	...	...	...	...
820	7203	0	45.0	30	instock	23.48	instock
821	7204	0	45.0	9	instock	24.18	instock
822	7247	1	54.8	6	instock	27.18	instock
823	7329	0	26.5	14	instock	13.42	instock
824	7338	1	16.3	40	instock	8.00	instock

825 rows × 7 columns

Entrée [32]:
```
#Vérification
df_erp.loc[df_erp['product_id'] == 4973]
```

Out[32]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	stock_status_2
449	4973	0	10.0	0	outofstock	4.96	instock

2.1.1.3 - Analyse de la variable ONSALE_WEB

Entrée [96]:
```
#Vérification de la colonne onsale_web et des valeurs qu'elle contient? Que signifient-elles?
print("Elle indique si un produit est en vente sur le site web")
```

```
Elle indique si un produit est en vente sur le site web
```

Entrée [94]:
```
#Quelles sont les colonnes à conserver selon vous?
print("Tout sauf la colonne 'stock_statuts2'")
```

```
Tout sauf la colonne 'stock_statuts2'
```

Entrée [35]:
```
#Supprimer la colonne comportant le libellé "stock_status_2" car elle est redondante
#avec la colonne "stock_status".
df_erp = df_erp.drop(columns=['stock_status_2'])
df_erp
```

Out[35]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price
0	3847	1	24.2	16	instock	12.88
1	3849	1	34.3	10	instock	17.54
2	3850	1	20.8	0	outofstock	10.64
3	4032	1	14.1	26	instock	6.92
4	4039	1	46.0	3	instock	23.77
...	...	...	...	...	...	...
820	7203	0	45.0	30	instock	23.48
821	7204	0	45.0	9	instock	24.18
822	7247	1	54.8	6	instock	27.18
823	7329	0	26.5	14	instock	13.42
824	7338	1	16.3	40	instock	8.00

825 rows × 6 columns

2.1.1.4 - Analyse de la variable prix d'achat

Entrée [36]:
```
######################
##   prix d'achat   ##
######################
```

Entrée [37]:
```
#Vérification de la colonne purchase_price :
```

Entrée [38]:
```
#Afficher le ou les prix non renseignés dans la colonne "purchase_price"
purchase_nul = df_erp[df_erp['purchase_price'].isnull()]
purchase_nul
```

Out[38]:

product_id	onsale_web	price	stock_quantity	stock_status	purchase_price

Entrée [39]:
```
#Afficher le prix minimum de la colonne "purchase_price"
purchase_min = df_erp['purchase_price'].min()
purchase_min
```

Out[39]: 2.74

Entrée [40]:
```
#Afficher le prix maximum de la colonne "purchase_price"
purchase_max = df_erp['purchase_price'].max()
purchase_max
```

Out[40]: 137.81

2.2 - Analyse exploratoire du fichier web.xlsx

Entrée [41]:
```
#Dimension du dataset
#Nombre d'observations

#Nombre de caractéristiques
print("les dimmensions du dataset erp sont :", df_web.shape)
```

```
les dimmensions du dataset erp sont : (1513, 29)
```

Entrée [42]:
```
#Consulter le nombre de colonnes
print("Le tableau web comporte {} colonne(s)".format(df_web.shape[1]))
#La nature des données dans chacune des colonnes
ndd_web = df_web.dtypes
print(ndd_web)
```

```
Le tableau web comporte 29 colonne(s)
sku                             object
virtual                          int64
downloadable                     int64
rating_count                     int64
average_rating                 float64
total_sales                    float64
tax_status                      object
tax_class                      float64
post_author                    float64
post_date                datetime64[ns]
post_date_gmt            datetime64[ns]
post_content                   float64
product_type                    object
post_title                      object
post_excerpt                    object
post_status                     object
comment_status                  object
ping_status                     object
post_password                  float64
post_name                       object
post_modified            datetime64[ns]
post_modified_gmt        datetime64[ns]
post_content_filtered          float64
post_parent                    float64
guid                            object
menu_order                     float64
post_type                       object
post_mime_type                  object
comment_count                  float64
dtype: object
```

Entrée [43]:
```
#Le nombre de valeurs présentes dans chacune des colonnes
ndv_web = df_web.count()
print(ndv_web)
```

```
sku                     1428
virtual                 1513
downloadable            1513
rating_count            1513
average_rating          1430
total_sales             1430
tax_status               716
tax_class                  0
post_author             1430
post_date               1430
post_date_gmt           1430
post_content               0
product_type            1429
post_title              1430
post_excerpt             716
post_status             1430
comment_status          1430
ping_status             1430
post_password              0
post_name               1430
post_modified           1430
post_modified_gmt       1430
post_content_filtered      0
post_parent             1430
guid                    1430
menu_order              1430
post_type               1430
post_mime_type           714
comment_count           1430
dtype: int64
```

Entrée [44]:
```
#Selon vous, quelles sont les colonnes à conserver ?
print("Toutes sauf les colonnes où il n'y a pas de valeurs")
```

```
Toutes sauf les colonnes où il n'y a pas de valeurs
```

Entrée [45]:
```
#Si vous avez défini des colonnes à supprimer, effectuer l'opération
colonne_sup = ['tax_class','post_content','post_password','post_content_filtered']
df_web = df_web.drop(columns=colonne_sup)
df_web.head()
```

Out[45]:

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_excerpt	p
0	11862	0	0	0	0.0	3.0	NaN	2.0	2018-02-12 13:46:23	2018-02-12 12:46:23	Vin	Gilles Robin Hermitage Rouge 2012	NaN	
1	16057	0	0	0	0.0	5.0	NaN	2.0	2018-04-17 15:29:17	2018-04-17 13:29:17	Vin	Domaine Pellé Sancerre Rouge La Croix Au Garde...	NaN	
2	14692	0	0	0	0.0	5.0	taxable	2.0	2019-03-19 10:06:47	2019-03-19 09:06:47	Vin	Château Fonréaud Bordeaux Blanc Le Cygne 2016	Grâce à la complémentarité des 3 cépages ...	
3	16295	0	0	0	0.0	14.0	NaN	2.0	2018-02-15 14:05:06	2018-02-15 13:05:06	Vin	Moulin de Gassac IGP Pays d'Hérault Guilhem Ro...	NaN	
4	15328	0	0	0	0.0	2.0	taxable	2.0	2019-03-27 18:05:09	2019-03-27 17:05:09	Vin	Agnès Levet Côte Rôtie Maestria 2017	<span style="float: none; background-color: tr...	

Entrée [46]:
```
#Visualisation des valeurs de la colonne sku
#Quelles sont les valeurs qui ne semblent pas respecter la régle de codification?
valeurs = df_web['sku'].unique()
valeurs
print("Les valeurs qui ne comporte pas des chiffres, comme : 13127-1,'bon-cadeau-25-euros','nan'")
```

```
Les valeurs qui ne comporte pas des chiffres, comme : 13127-1,'bon-cadeau-25-euros','nan'
```

Entrée [47]:
```
#Suppression des lignes où les valeurs sont nulles
df_web = df_web.dropna(subset=['sku'])
df_web.head()
```

Out[47]:

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_excerpt	p
0	11862	0	0	0	0.0	3.0	NaN	2.0	2018-02-12 13:46:23	2018-02-12 12:46:23	Vin	Gilles Robin Hermitage Rouge 2012	NaN	
1	16057	0	0	0	0.0	5.0	NaN	2.0	2018-04-17 15:29:17	2018-04-17 13:29:17	Vin	Domaine Pellé Sancerre Rouge La Croix Au Garde...	NaN	
2	14692	0	0	0	0.0	5.0	taxable	2.0	2019-03-19 10:06:47	2019-03-19 09:06:47	Vin	Château Fonréaud Bordeaux Blanc Le Cygne 2016	Grâce à la complémentarité des 3 cépages ...	
3	16295	0	0	0	0.0	14.0	NaN	2.0	2018-02-15 14:05:06	2018-02-15 13:05:06	Vin	Moulin de Gassac IGP Pays d'Hérault Guilhem Ro...	NaN	
4	15328	0	0	0	0.0	2.0	taxable	2.0	2019-03-27 18:05:09	2019-03-27 17:05:09	Vin	Agnès Levet Côte Rôtie Maestria 2017	<span style="float: none; background-color: tr...	

Entrée [48]:
```
#Si vous avez identifié des codes articles ne respectant pas la régle de codification, consultez-les?
filtre = df_web['sku'].astype(str).str.isdigit() == False
no_regle = df_web[filtre]
no_regle.head(100)
```

Out[48]:

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_exc
272	13127-1	0	0	0	0.0	4.0	taxable	2.0	2020-06-09 15:42:04	2020-06-09 13:42:04	Vin	Clos du Mont-Olivet Châteauneuf-du-Pape 2007	Nez grac très élé avec une to
842	bon-cadeau-25-euros	0	0	0	0.0	7.0	NaN	1.0	2018-06-01 13:53:46	2018-06-01 11:53:46	Autre	Bon cadeau de 25€	
1117	13127-1	0	0	0	0.0	4.0	NaN	2.0	2020-06-09 15:42:04	2020-06-09 13:42:04	Vin	Clos du Mont-Olivet Châteauneuf-du-Pape 2007	
1387	bon-cadeau-25-euros	0	0	0	0.0	7.0	taxable	1.0	2018-06-01 13:53:46	2018-06-01 11:53:46	NaN	Bon cadeau de 25€	< style="c #a852 Pa

Entrée [49]:
```
#Identifier les lignes sans code articles
sans_code = df_web[df_web['sku'].isnull()]
sans_code
```

Out[49]:

sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_excerpt	post_st

Entrée [50]:
```
#Pour les codes articles identifiés, réalisé une analyse et définissez l'action à entreprendre
df_web_ok = df_web['sku'].astype(str).str.isdigit()
df_web[df_web_ok]
```

Out[50]:

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_excer
0	11862	0	0	0	0.0	3.0	NaN	2.0	2018-02-12 13:46:23	2018-02-12 12:46:23	Vin	Gilles Robin Hermitage Rouge 2012	Na
1	16057	0	0	0	0.0	5.0	NaN	2.0	2018-04-17 15:29:17	2018-04-17 13:29:17	Vin	Domaine Pellé Sancerre Rouge La Croix Au Garde...	Na
2	14692	0	0	0	0.0	5.0	taxable	2.0	2019-03-19 10:06:47	2019-03-19 09:06:47	Vin	Château Fonréaud Bordeaux Blanc Le Cygne 2016	Grâce à complémentari des 3 cépage
3	16295	0	0	0	0.0	14.0	NaN	2.0	2018-02-15 14:05:06	2018-02-15 13:05:06	Vin	Moulin de Gassac IGP Pays d'Hérault Guilhem Ro...	Na
4	15328	0	0	0	0.0	2.0	taxable	2.0	2019-03-27 18:05:09	2019-03-27 17:05:09	Vin	Agnès Levet Côte Rôtie Maestria 2017	<spa style="floa non backgroun color: tr
...	...	...	...	...	...	...	...	...	...	...	...	...	
1508	16326	0	0	0	0.0	5.0	taxable	2.0	2019-04-18 11:32:46	2019-04-18 09:32:46	Vin	Camin Larredya Jurançon Moelleux Au Capcéu 2018	Sur le millésim 2017, A Capceu c domaine Ca
1509	15662	0	0	0	0.0	15.0	taxable	2.0	2018-02-27 10:13:03	2018-02-27 09:13:03	Vin	Chermette Domaine du Vissoux Beaujolais Griott...	C'est Beaujola typique : fruit frais, g
1510	15329	0	0	0	0.0	3.0	NaN	2.0	2019-03-27 18:28:15	2019-03-27 17:28:15	Vin	Agnès Levet Côte Rôtie Péroline 2017	Na

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_excer
1511	14827	0	0	0	0.0	7.0	NaN	2.0	2018-11-26 09:56:52	2018-11-26 08:56:52	Vin	Marc Colin Et Fils Chassagne-Montrachet Blanc ...	Na
1512	16004	0	0	0	0.0	5.0	NaN	2.0	2018-06-07 16:27:25	2018-06-07 14:27:25	Vin	Château du Couvent Pomerol 2017	Na

1424 rows × 25 columns

Entrée [51]:
```
#La clé pour chaque ligne est-elle uniques? ou autrement dit, y a-t-il des doublons?
doublons = df_web['sku'].duplicated()
nombre_doublons = doublons.sum()
print(nombre_doublons)
```

```
714
```

Entrée [52]:
```
df_web =df_web[df_web['post_type'] == 'product']
df_web
```

Out[52]:

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_e
2	14692	0	0	0	0.0	5.0	taxable	2.0	2019-03-19 10:06:47	2019-03-19 09:06:47	Vin	Château Fonréaud Bordeaux Blanc Le Cygne 2016	Grâ compléme des 3 cépa
4	15328	0	0	0	0.0	2.0	taxable	2.0	2019-03-27 18:05:09	2019-03-27 17:05:09	Vin	Agnès Levet Côte Rôtie Maestria 2017	<span style none; backg co
6	16515	0	0	0	0.0	10.0	taxable	2.0	2018-06-02 09:31:31	2018-06-02 07:31:31	Vin	Château Turcaud Bordeaux Rouge Cuvée Majeure 2018	id="wrapper"> id="cor wra
11	16585	0	0	0	0.0	15.0	taxable	2.0	2018-02-16 14:03:16	2018-02-16 13:03:16	Vin	Xavier Frissant Touraine Sauvignon 2019	Un joli sau frais et m avec
14	12869	0	0	0	0.0	7.0	taxable	2.0	2019-03-28 14:29:35	2019-03-28 13:29:35	Vin	Stéphane Tissot Arbois D.D. 2016	Un Vin co éclatant. Le n fr
...	...	...	...	...	...	...	...	...	...	...	...	...	
1503	13074	0	0	0	0.0	4.0	taxable	2.0	2018-02-12 14:25:28	2018-02-12 13:25:28	Vin	Château de Vaudieu Châteauneuf-du-Pape L'Avenu...	"L'Aven issue d'une p de vie
1505	16322	0	0	0	0.0	0.0	taxable	2.0	2018-02-15 13:51:32	2018-02-15 12:51:32	Vin	Moulin de Gassac IGP Pays d'Hérault Guilhem Ro...	Belle com aromatique fruit
1507	12365	0	0	0	0.0	10.0	taxable	2.0	2019-01-29 15:53:05	2019-01-29 14:53:05	Vin	Parés Baltà Penedès Electio 2013	Une cuvée p avec une très
1508	16326	0	0	0	0.0	5.0	taxable	2.0	2019-04-18 11:32:46	2019-04-18 09:32:46	Vin	Camin Larredya Jurançon Moelleux Au Capcéu 2018	Sur le mi 2017, Au C du domain

	sku	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_author	post_date	post_date_gmt	product_type	post_title	post_e
1509	15662	0	0	0	0.0	15.0	taxable	2.0	2018-02-27 10:13:03	2018-02-27 09:13:03	Vin	Chermette Domaine du Vissoux Beaujolais Griott...	C'est le Bea typique fra

714 rows × 25 columns

Entrée [53]:
```
#Les lignes sans code article semble être toutes non renseignés
#Pour s'en assurer réaliser les étapes suivantes:
#1 - Créer un dataframe avec uniquement les lignes sans code article
sans_code
#2 - utiliser la fonction df.info() sur ce nouveau dataframe pour observer le nombre de valeur reseigner dans chacune des colonnes
sans_code.info
#3 - Que constatez-vous?
print("Elles sont toutes nulles")
```

```
Elles sont toutes nulles
```

2.3 - Analyse exploratoire du fichier liaison.xlsx

Entrée [54]:
```
#Dimension du dataset

#Nombre d'observations

#Nombre de caractéristiques
print("les dimmensions du dataset erp sont :", df_liaison.shape)
```

```
les dimmensions du dataset erp sont : (825, 2)
```

Entrée [55]:
```
#Consulter le nombre de colonnes
print("Le tableau liaison comporte {} colonne(s)".format(df_liaison.shape[1]))
#La nature des données dans chacune des colonnes
ndd_liaison = df_liaison.dtypes
print(ndd_liaison)
```

```
Le tableau liaison comporte 2 colonne(s)
id_web        object
product_id     int64
dtype: object
```

Entrée [56]:
```
#Le nombre de valeurs présentes dans chacune des colonnes
ndv_liaison = df_liaison.count()
print(ndv_liaison)
```

```
id_web        734
product_id    825
dtype: int64
```

Entrée [57]:
```
#Les valeurs de la colonne "product_id" sont elles toutes uniques?
doublons_liaison = df_liaison['product_id'].duplicated()
nombre_doublons_liaison = doublons_liaison.sum()
print(nombre_doublons_liaison)
print('Elles sont toutes uniques')
```

```
0
Elles sont toutes uniques
```

Entrée [58]:
```
#Les valeurs de la colonne "id_web" sont-elles toutes uniques?
doublons_liaison2 = df_liaison['id_web'].duplicated()
nombre_doublons_liaison2 = doublons_liaison2.sum()
print(nombre_doublons_liaison)
print('Elles sont toutes uniques')
```

```
0
Elles sont toutes uniques
```

Entrée [59]:
```
#Avons-nous des articles sans correspondances?
print('Oui car il y a un ecart de 91 valeurs')
```

```
Oui car il y a un ecart de 91 valeurs
```

Etape 3 - Jonction des fichiers

Etape 3.1 - Jonction du fichier df_erp et df_liaison

Entrée [60]:
```
#Fusion des fichiers df_erp et df_liaison
df_merge = df_erp.merge(df_liaison, how='outer', on='product_id')
df_merge
```

Out[60]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web
0	3847	1	24.2	16	instock	12.88	15298
1	3849	1	34.3	10	instock	17.54	15296
2	3850	1	20.8	0	outofstock	10.64	15300
3	4032	1	14.1	26	instock	6.92	19814
4	4039	1	46.0	3	instock	23.77	19815
...	...	...	...	...	...	...	...
820	7203	0	45.0	30	instock	23.48	NaN
821	7204	0	45.0	9	instock	24.18	NaN
822	7247	1	54.8	6	instock	27.18	13127-1
823	7329	0	26.5	14	instock	13.42	14680-1
824	7338	1	16.3	40	instock	8.00	16230

825 rows × 7 columns

Entrée [61]:
```
#Y a t-il des lignes ne "matchant" entre les 2 fichiers?
absent = df_merge[df_merge['product_id'].isnull()]
absent.head()
```

Out[61]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web

Etape 3.2 - Jonction du fichier df_merge et df_web

Entrée [62]:
```
#Fusionnez les datasets df_merge et df_web
df_web.rename(columns={'sku':'id_web'},inplace=True)
fusion = df_merge.merge(df_web, how='outer', on='id_web')
fusion.head(5)
fusion.shape
```

Out[62]: (825, 31)

Entrée [63]:
```
#Avons-nous des lignes sans correspondances?
ndv_fusion = fusion.count()
ndv_fusion
```

Out[63]:
```
product_id           825
onsale_web           825
price                825
stock_quantity       825
stock_status         825
purchase_price       825
id_web               734
virtual              714
downloadable         714
rating_count         714
average_rating       714
total_sales          714
tax_status           714
post_author          714
post_date            714
post_date_gmt        714
product_type         713
post_title           714
post_excerpt         714
post_status          714
comment_status       714
ping_status          714
post_name            714
post_modified        714
post_modified_gmt    714
post_parent          714
guid                 714
menu_order           714
post_type            714
post_mime_type         0
comment_count        714
dtype: int64
```

Entrée [64]:
```
absent2 = fusion[fusion['product_id'].isnull()]
absent2.head()
```

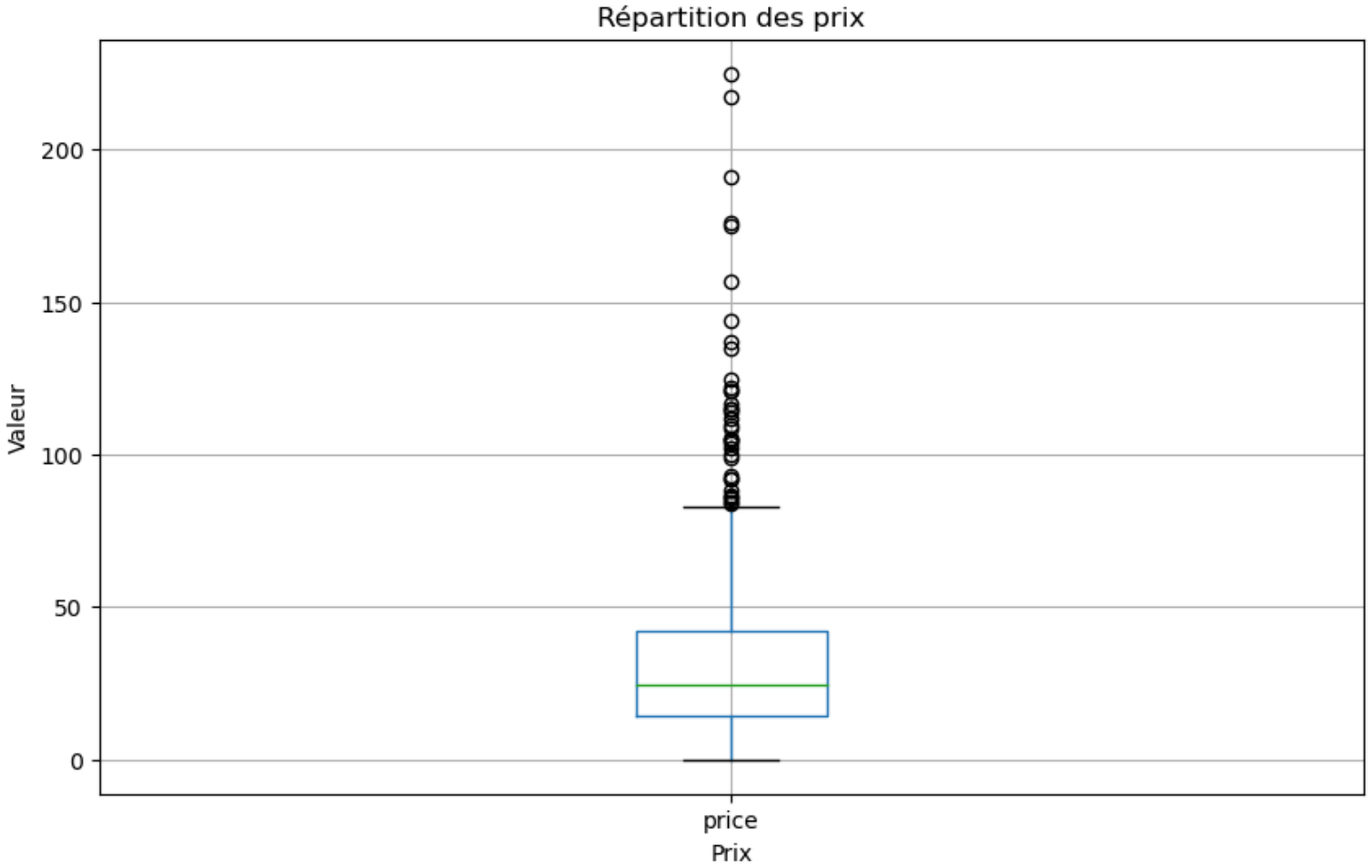
Out[64]:

product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post_

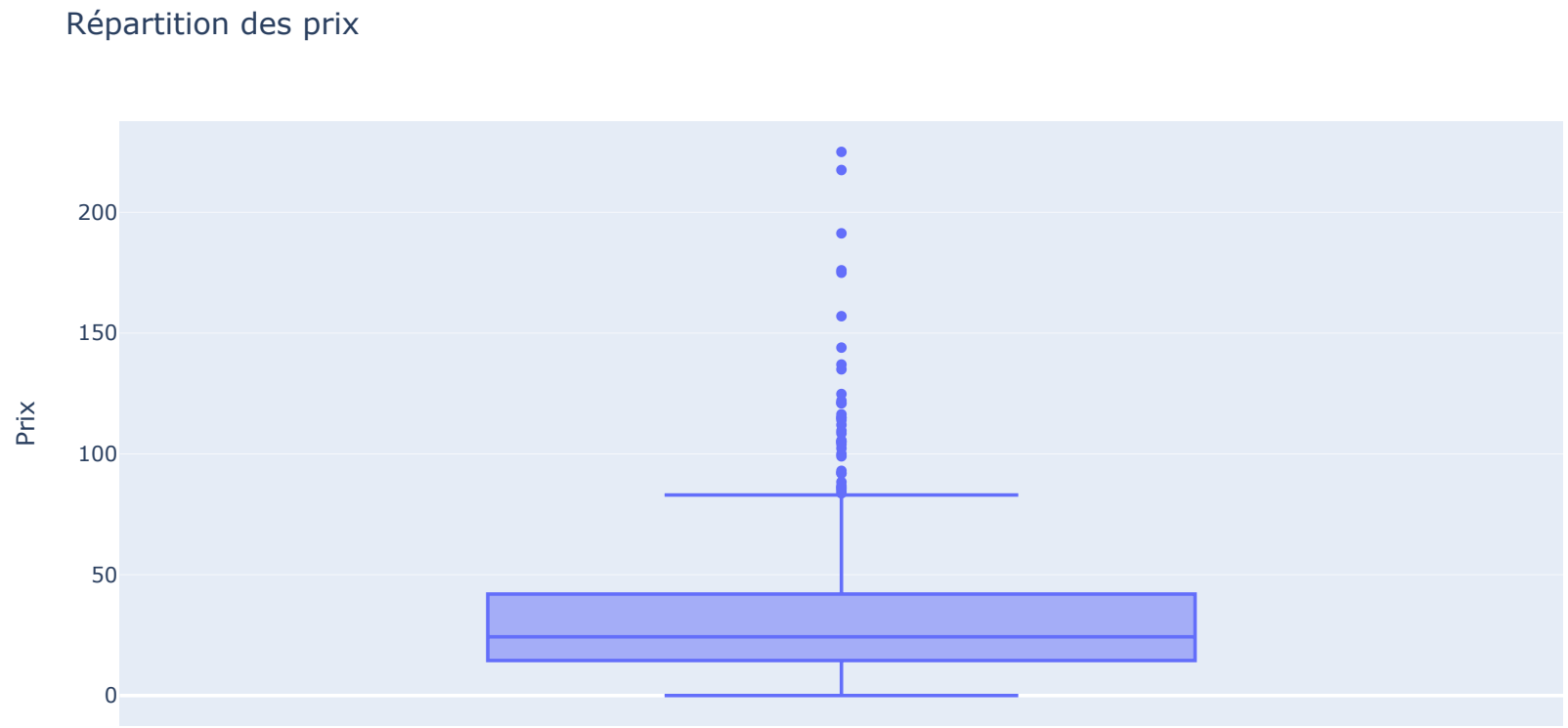
Etape 4 - Analyse univarié des prix

Etape 4.1 - Exploration par la visualisation de données

Entrée [65]:
```
#Création d'une Boite à moustache de la répartition des prix grâce à Pandas
import matplotlib.pyplot as plt
plt.figure(figsize=(10,6))
fusion.boxplot(column='price')
plt.title('Répartition des prix')
plt.xlabel('Prix')
plt.ylabel('Valeur')
plt.show()
```


Répartition des prix
Valeur
200
150
100
50
0
price
Prix

Entrée [66]:
```
#Autre méthode avec plotly express
import plotly.express as px
fig = px.box(fusion, y='price', title ='Répartition des prix', labels={'price':'Prix'})
fig.show()
```



Etape 4.2 - Exploration par l'utisation de méthodes statistique

Etape 4.2.1 - Identification par le Z-index

Entrée [67]:
```
#Calculer la moyenne du prix
moyenne = fusion['price'].mean()
print('La moyenne est de :',moyenne)
#Calculer l'écart-type du prix
ecart_type = fusion['price'].std()
print("L'écart type est de :",ecart_type)
#Calculer le Z-score
fusion['z_score'] = (fusion['price'] - moyenne) / ecart_type
fusion[['id_web', 'price','z_score']]
```

```
La moyenne est de : 32.23266666666667
L'écart type est de : 26.645280282385123
```

Out[67]:

	id_web	price	z_score
0	15298	24.2	-0.301467
1	15296	34.3	0.077587
2	15300	20.8	-0.429069
3	19814	14.1	-0.680521
4	19815	46.0	0.516689
...	...	...	...
820	15891	27.5	-0.177617
821	15887	69.0	1.379882
822	13127-1	54.8	0.846954
823	14680-1	26.5	-0.215148
824	16230	16.3	-0.597955

825 rows × 3 columns

Entrée [68]:
```
#Quel est le seuil prix dont z-score est supérieur à 3?
seuil = fusion[fusion['z_score'] >3]
seuil['price'].min()
```

Out[68]: 114.0

Etape 4.2.2 - Identification par l'interval interquartile

Entrée [69]:
```
#Utilisation de la fonction describe de Pandas pour l'etude des mesures de dispersions
describe = fusion.describe()
describe
```

Out[69]:

	product_id	onsale_web	price	stock_quantity	purchase_price	virtual	downloadable	rating_count	average_rating	total_sales	post_author	post_
count	825.000000	825.000000	825.000000	825.000000	825.000000	714.0	714.0	714.0	714.0	714.000000	714.000000	
mean	5162.597576	0.867879	32.232667	21.602424	16.940582	0.0	0.0	0.0	0.0	8.054622	1.998599	2018-0 03:57:52.95098
min	3847.000000	0.000000	0.000000	0.000000	2.740000	0.0	0.0	0.0	0.0	0.000000	1.000000	2018-0 12:5
25%	4348.000000	1.000000	14.500000	7.000000	7.590000	0.0	0.0	0.0	0.0	5.000000	2.000000	2018-0 20:01:12.50
50%	4907.000000	1.000000	24.300000	18.000000	12.710000	0.0	0.0	0.0	0.0	8.000000	2.000000	2018-0 14:5
75%	5805.000000	1.000000	42.000000	30.000000	22.020000	0.0	0.0	0.0	0.0	11.000000	2.000000	2019-0 14:3
max	7338.000000	1.000000	225.000000	145.000000	137.810000	0.0	0.0	0.0	0.0	36.000000	2.000000	2020-0 11:0
std	902.644635	0.338828	26.645280	21.917863	14.561840	0.0	0.0	0.0	0.0	4.161344	0.037424	

Entrée [70]:
```
#Définissez un seuil pour les articles "outliers" en prix
q1 = fusion['price'].quantile(0.25)
q3 = fusion['price'].quantile(0.75)
iqr = q3-q1
borne_sup = q3+1.5*iqr
print(f"Borne supérieur : {borne_sup}")
outliers = fusion[(fusion['price'] > borne_sup)]
outliers
```

530	5007	1	105.0	15	instock	55.88	12791	0.0	0.0	0.0	0.0	3.0	taxable
531	5008	1	105.0	12	instock	56.42	11602	0.0	0.0	0.0	0.0	7.0	taxable
538	5025	1	112.0	136	instock	68.60	13914	0.0	0.0	0.0	0.0	6.0	taxable
539	5026	1	86.8	101	instock	50.13	13913	0.0	0.0	0.0	0.0	9.0	taxable

Entrée [71]:
```
#Définissez le nombre d'articles et la proportion de l'ensemble du catalogue "outliers"
nombre = outliers.shape[0]
print("Nombre d'articles outliers :",nombre)
total = fusion.shape[0]
proportion = nombre / total * 100
print("La proportion d'outliers est de :", proportion)
```

```
Nombre d'articles outliers : 36
La proportion d'outliers est de : 4.363636363636364
```

Entrée [92]:
```
#Selon vous, ces outliers sont-ils justifiés ? Comment le démontrer si cela est possible ?
print("Oui ils sont justifiés car nous avons des produits d'exception")
```

```
Oui ils sont justifiés car nous avons des produits d'exception
```

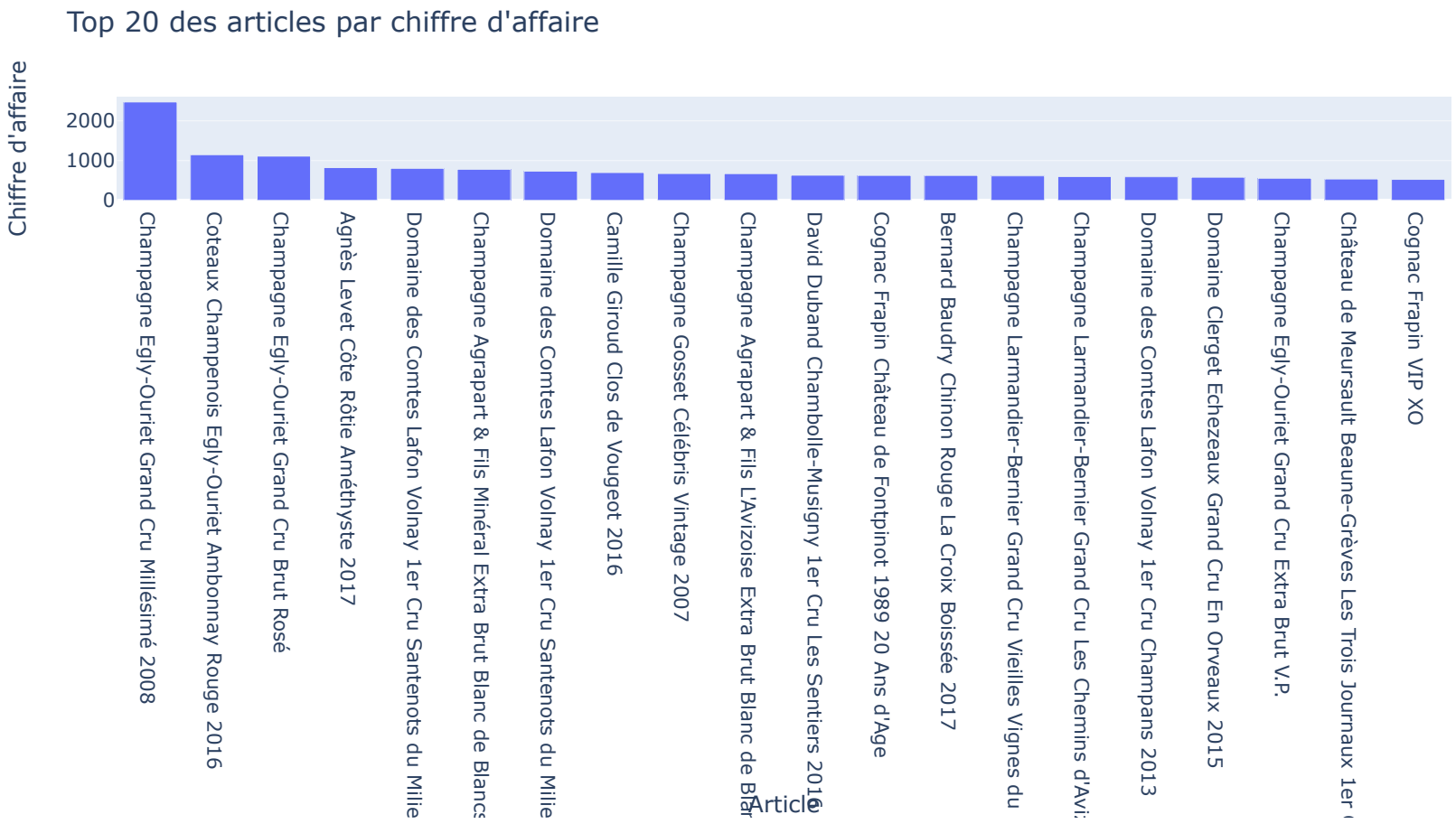
Etape 5 - Analyse univarié du CA, des quantités vendues, des stocks et de la marge ainsi qu'une analyse multivarié

Etape 5.1 - Analyse des ventes en CA

Entrée [73]:
```
##############################
# Calculer le CA su site web #
##############################

#Créez une colonne calculant le CA par article

#Calculez la somme de la colonne "ca_par_article"
fusion['ca_par_article']=fusion['price']*fusion['total_sales']
#Ce résultat correspond au chiffre d'affaire du site web
ca_fusion = fusion['ca_par_article'].sum()
print("Le chiffre d'affaire est de :",ca_fusion)
```

```
Le chiffre d'affaire est de : 143680.1
```

Entrée [74]:
```
################################
# Palmares des articles en CA #
################################

#Effectuer le tri dans l'ordre décroissant du CA du dataset df_merge
fusion = fusion.sort_values(by='ca_par_article', ascending=False)
#Réinitialiser l'index du dataset par un reset_index
fusion = fusion.reset_index(drop=True)
#Afficher les 20 premier articles en CA
decroi_t20_ca = fusion.sort_values(by='ca_par_article', ascending=False)
top_20_ca = decroi_t20_ca.head(20)
#Graphique en barre des 20 premiers articles avec plotly express
graphique = px.bar(top_20_ca,
                   x='post_title',
                   y='ca_par_article',
                  title = "Top 20 des articles par chiffre d'affaire",
                  labels={'post_title':'Article',
                          'ca_par_article':"Chiffre d'affaire"})
graphique.show()
```


Top 20 des articles par chiffre d'affaire
Chiffre d'affaire
2000
1000
0
Champagne Egly-Ouriet Grand Cru Millésimé 2008
Coteaux Champenois Egly-Ouriet Ambonnay Rouge 2016
Champagne Egly-Ouriet Grand Cru Brut Rosé
Agnès Levet Côte Rôtie Améthyste 2017
Domaine des Comtes Lafon Volnay 1er Cru Santenots du Milie
Champagne Agrapart & Fils Minéral Extra Brut Blanc de Blancs
Domaine des Comtes Lafon Volnay 1er Cru Santenots du Milie
Camille Giroud Clos de Vougeot 2016
Champagne Gosset Célébris Vintage 2007
Champagne Agrapart & Fils L'Avizoise Extra Brut Blanc de Bl
David Duband Chambolle-Musigny 1er Cru Les Sentiers 2016
Cognac Frapin Château de Fontpinot 1989 20 Ans d'Age
Bernard Baudry Chinon Rouge La Croix Boissée 2017
Champagne Larmandier-Bernier Grand Cru Vieilles Vignes du
Champagne Larmandier-Bernier Grand Cru Les Chemins d'Avi
Domaine des Comtes Lafon Volnay 1er Cru Champans 2013
Domaine Clerget Echezeaux Grand Cru En Orveaux 2015
Champagne Egly-Ouriet Grand Cru Extra Brut V.P.
Château de Meursault Beaune-Grèves Les Trois Journaux 1er
Cognac Frapin VIP XO
Article

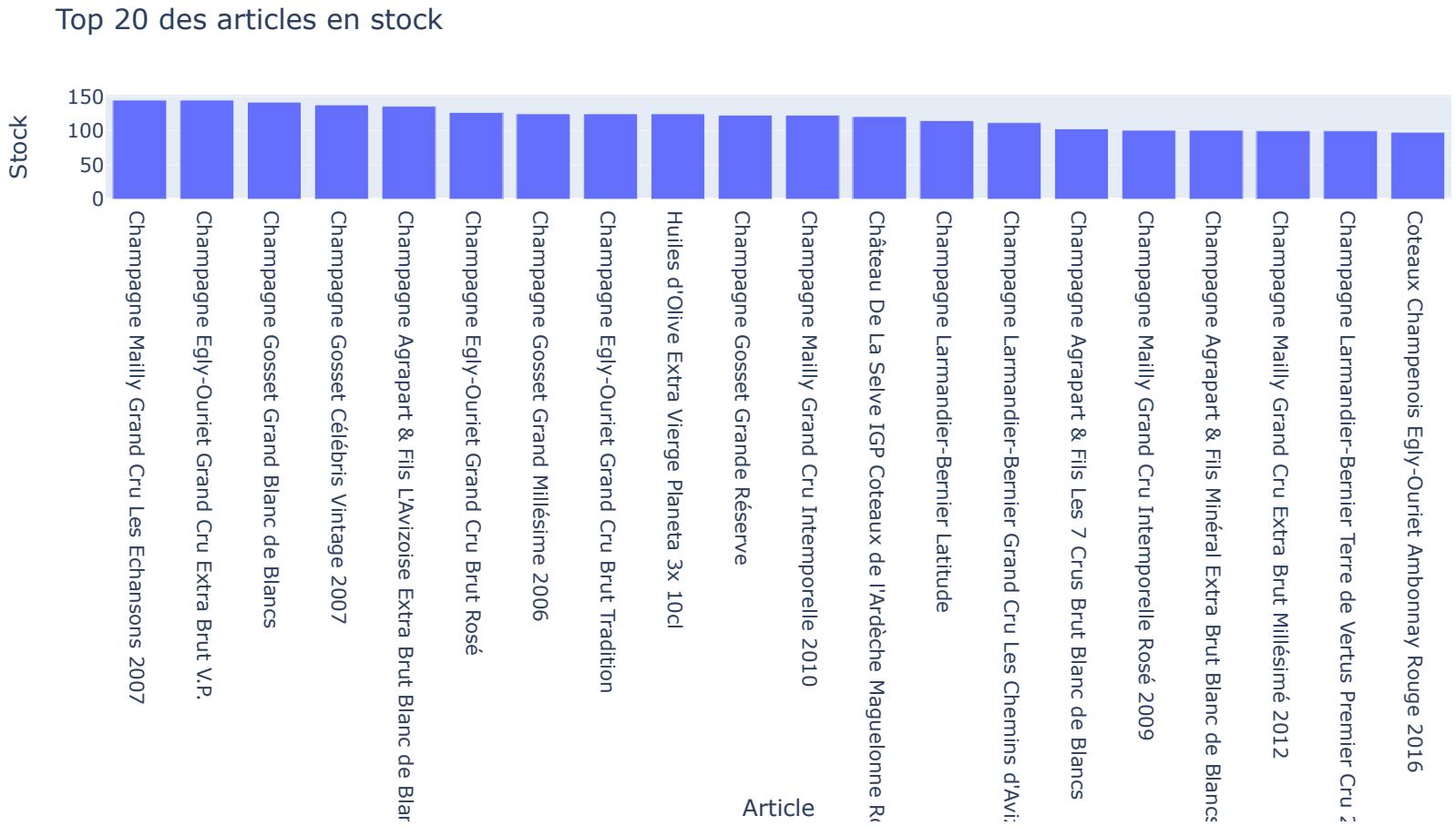
Entrée [75]:
```
##############################
# Calculer le 20 / 80 en CA #
##############################

#Créer une colonne calculant la part du CA de la ligne dans le dataset
fusion['part_du_ca'] = fusion['ca_par_article'] / ca_fusion
#Créer une colonne réalisant la somme cumulative de la colonne précedemment créée
fusion['cumul_somme'] = fusion['part_du_ca'].cumsum()
#Grâce au deux colonnes créées précedemment, calculer le nombre d'articles représentant 80% du CA
nb_article80p = fusion[fusion['cumul_somme'] <= 0.8].index[-1]+1
print("Le nombre d'articles représentant 80% du CA est:",nb_article80p)
#Afficher la proportion que représentent ce groupe d'articles dans le catalogue entier du site web
total_article = len(fusion)
proportion_80 = nb_article80p / total_article
print("La proportion des 80% du ca représente :", proportion_80 )
```

```
Le nombre d'articles représentant 80% du CA est: 434
La proportion des 80% du ca représente : 0.526060606060606
```

Etape 5.2 - Analyse des ventes en Quantités

Entrée [76]:

```
#####################################
# Palmares des articles en quantité #
#####################################

#Effectuer le tri dans l'ordre décroissant de quantités vendues du dataset df_merge
fusion = fusion.sort_values(by='stock_quantity', ascending=False)
#Réinitialiser l'index du dataset par un reset_index
fusion = fusion.reset_index(drop=True)
#Afficher les 20 premier articles en quantité
decroi_t20_qt = fusion.sort_values(by='stock_quantity', ascending=False)
top_20_qt = decroi_t20_qt.head(20)
top_20_qt
#Graphique en barre des 20 premiers articles avec plotly express
graphique = px.bar(top_20_qt,
                   x='post_title',
                   y='stock_quantity',
                  title = "Top 20 des articles en stock",
                  labels={'post_title':'Article',
                          'stock_quantity':"Stock"})
graphique.show()
#Réinitialiser l'index du dataset par un reset_index
fusion = fusion.reset_index(drop=True)
```


Top 20 des articles en stock
Stock
150
100
50
0
Champagne Mailly Grand Cru Les Echansons 2007
Champagne Egly-Ouriet Grand Cru Extra Brut V.P.
Champagne Gosset Grand Blanc de Blancs
Champagne Gosset Célébris Vintage 2007
Champagne Agrapart & Fils L'Avizoise Extra Brut Blanc de Blar
Champagne Egly-Ouriet Grand Cru Brut Rosé
Champagne Gosset Grand Millésime 2006
Champagne Egly-Ouriet Grand Cru Brut Tradition
Huiles d'Olive Extra Vierge Planeta 3x 10cl
Champagne Gosset Grande Réserve
Champagne Mailly Grand Cru Intemporelle 2010
Château De La Selve IGP Coteaux de l'Ardèche Maguelonne R
Champagne Larmandier-Bernier Latitude
Champagne Larmandier-Bernier Grand Cru Les Chemins d'Avi
Champagne Agrapart & Fils Les 7 Crus Brut Blanc de Blancs
Champagne Mailly Grand Cru Intemporelle Rosé 2009
Champagne Agrapart & Fils Minéral Extra Brut Blanc de Blancs
Champagne Mailly Grand Cru Extra Brut Millésimé 2012
Champagne Larmandier-Bernier Terre de Vertus Premier Cru
Coteaux Champenois Egly-Ouriet Ambonnay Rouge 2016
Article

Entrée [77]:
```
#Afficher les 20 premier articles en CA
top_20_ca.head(20)
```

Out[77]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	pos
0	4352	1	225.0	0	outofstock	137.81	15940	0.0	0.0	0.0	0.0	11.0	taxable	
1	5892	1	191.3	98	instock	116.06	14983	0.0	0.0	0.0	0.0	6.0	taxable	
2	4353	1	79.5	127	instock	45.91	12587	0.0	0.0	0.0	0.0	14.0	taxable	
3	5826	1	41.2	34	instock	21.71	15325	0.0	0.0	0.0	0.0	20.0	taxable	
4	6212	1	115.0	16	instock	59.42	13996	0.0	0.0	0.0	0.0	7.0	taxable	
5	5026	1	86.8	101	instock	50.13	13913	0.0	0.0	0.0	0.0	9.0	taxable	
6	5008	1	105.0	12	instock	56.42	11602	0.0	0.0	0.0	0.0	7.0	taxable	
7	5767	1	175.0	12	instock	90.42	15185	0.0	0.0	0.0	0.0	4.0	taxable	
8	6126	1	135.0	138	instock	80.33	14923	0.0	0.0	0.0	0.0	5.0	taxable	

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	pos
9	5025	1	112.0	136	instock	68.60	13914	0.0	0.0	0.0	0.0	6.0	taxable	
10	6201	1	105.6	16	instock	57.29	14596	0.0	0.0	0.0	0.0	6.0	taxable	
11	4406	1	157.0	12	instock	69.08	7819	0.0	0.0	0.0	0.0	4.0	taxable	
12	4647	1	28.5	45	instock	14.14	16525	0.0	0.0	0.0	0.0	22.0	taxable	
13	4358	1	77.0	81	instock	47.16	13854	0.0	0.0	0.0	0.0	8.0	taxable	
14	4359	1	85.6	112	instock	51.93	13853	0.0	0.0	0.0	0.0	7.0	taxable	
15	6214	1	99.0	9	instock	49.62	11601	0.0	0.0	0.0	0.0	6.0	taxable	
16	6202	1	116.4	12	instock	63.15	15126	0.0	0.0	0.0	0.0	5.0	taxable	

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	pos
17	4350	1	79.5	145	instock	47.30	12588	0.0	0.0	0.0	0.0	7.0	taxable	
18	4573	1	67.2	12	instock	36.46	13604	0.0	0.0	0.0	0.0	8.0	taxable	
19	4402	1	176.0	11	instock	78.25	3510	0.0	0.0	0.0	0.0	3.0	taxable	

Entrée [78]:
```
##############################
# Calculer le 20 / 80 en CA #
##############################

#Créer une colonne calculant la part en quantité de la ligne dans le dataset
qt_tot = fusion['stock_quantity'].sum()
fusion['part_qt'] = fusion['stock_quantity'] / qt_tot
#Créer une colonne réalisant la somme cumulative de la colonne précedemment créée
fusion['cumul_qt'] = fusion['part_qt'].cumsum()
#Grâce au deux colonnes créées précedemment, calculer le nombre d'articles représentant 80% des ventes en quantité
nb_article_80qt = fusion[fusion['cumul_qt'] <= 0.8].index[-1]+1
print("Le nombre d'articles représentant 80% du CA est:",nb_article_80qt)
#Afficher la proportion que représentent ce groupe d'articles dans le catalogue entier du site web
total_article = len(fusion)
proportion_80_qt = nb_article_80qt / total_article
print("La proportion des 80% du ca représente :", proportion_80_qt )
```

```
Le nombre d'articles représentant 80% du CA est: 378
La proportion des 80% du ca représente : 0.4581818181818182
```

Etape 5.3 - Analyse des stocks

Entrée [79]:

```
#######################################
# Calcule le nombre de mois de stock #
#######################################

#Import de numpy

#Création de la colonne Rotation de stock
fusion['rotation_stock'] = fusion['stock_quantity'] / fusion['total_sales']
#Remplacement des "inf" par 0
fusion['rotation_stock'].replace([np.inf, -np.inf],0 , inplace=True)
#Effectuer le tri dans l'ordre décroissant du nombre de mois de stock dans le dataset df_merge
fusion_rotation = fusion.sort_values(by='rotation_stock', ascending = False)
#Graphique en barre du flop 20 des produits qui ont le plus de mois de stock
flop_20_rotation = fusion_rotation.head(20)
graphique2 = px.bar(flop_20_rotation,
                    x='post_title',
                    y='rotation_stock',
                   title = "Flop 20 des produits qui ont le plus de mois de stock",
                   labels={'post_title':'Produit',
                           'rotation_stock':"Rotation_Stock"})
graphique2.show()
```

Entrée [80]:
```
####################################
# Valorisation des stocks en euros #
####################################

#Création de la colonne Valorisation des stocks en euros
fusion['valorisation_stock'] = fusion['stock_quantity']*fusion['price']
#Calculer la somme de la colonne "Valorisation_stock_euros"
Valorisation_stock_euros = fusion['valorisation_stock'].sum()
print("La valorisation des stock en euro est de :",Valorisation_stock_euros)
```

```
La valorisation des stock en euro est de : 531946.2
```

Entrée [81]:
```
##############################################
# Valorisation du nombre de produit en stock #
##############################################

#Calculer la somme de la colonne stock quantity
somme_qt = fusion['stock_quantity'].sum()
print("La somme totale est de :", somme_qt)
```

```
La somme totale est de : 17822
```

Etape 5.4 - Analyse du taux de marge

Entrée [93]:
```
############################
# Analyse du taux de marge #
############################
tva = 0.2
#Création de la colonne prix HT
fusion['prix_ht'] = fusion['price'] / (1 + tva)
#Création de la colonne Taux de marge
fusion['taux_marge'] = ((fusion['price'] - fusion['purchase_price']) / fusion['purchase_price'])*100
#Afficher le prix minimum de la colonne "taux_marge"
tm_min = fusion['taux_marge'].min()
print("Le taux de marge minimun est de :", tm_min)
#Afficher le prix maximum de la colonne "taux_marge"
tm_max = fusion['taux_marge'].max()
print("Le taux de marge maximun est de :", tm_max)
```

```
Le taux de marge minimun est de : -100.0
Le taux de marge maximun est de : 129.6949650863653
```

Entrée [83]:
```
#affichage de la ligne avec un taux de marge inférieur à 0
tm_neg = fusion[fusion['taux_marge'] < 0]
tm_neg
```

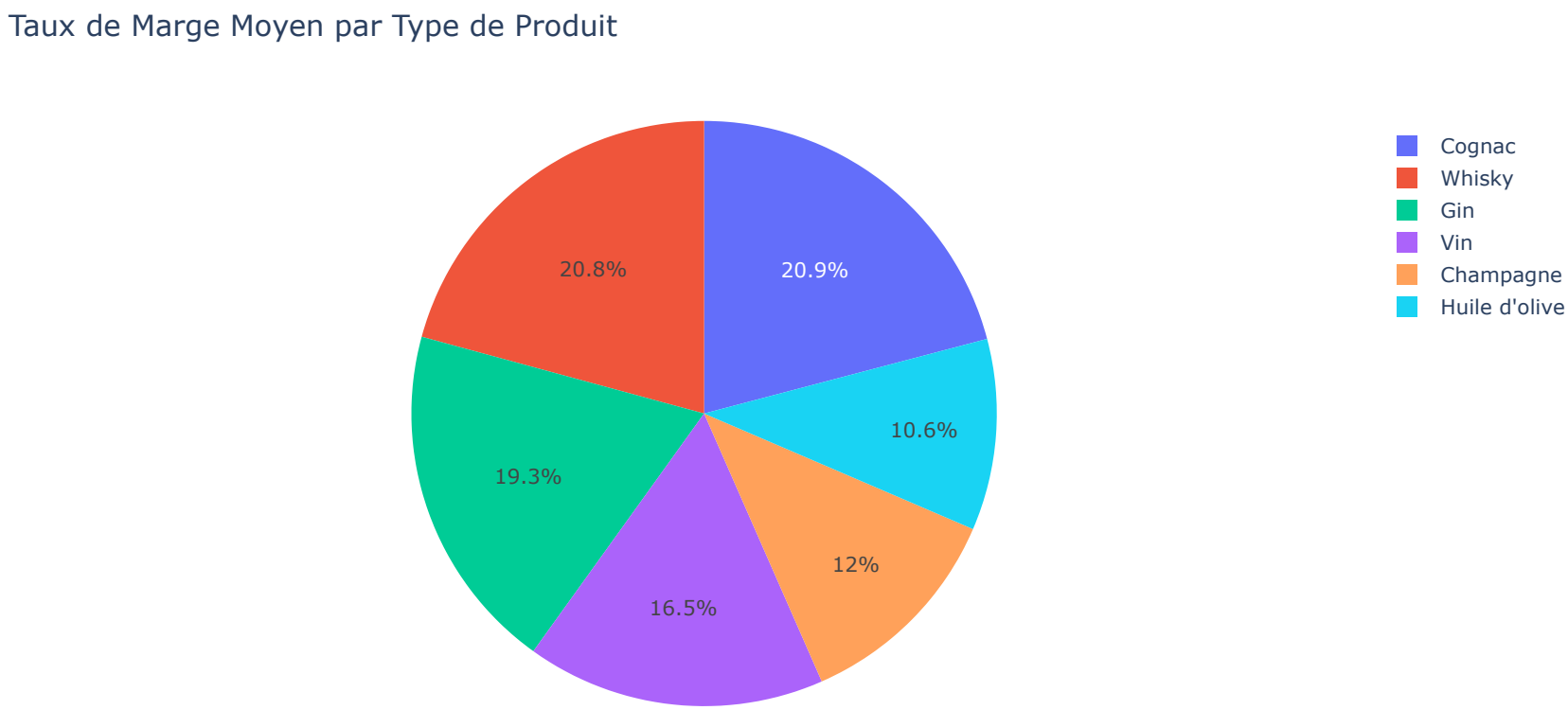
Out[83]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	pc
20	4355	1	12.65	97	instock	77.48	12589	0.0	0.0	0.0	0.0	0.0	taxable	
34	7196	0	31.00	55	instock	31.20	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
390	6594	0	0.00	19	outofstock	4.61	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
413	6324	0	92.00	18	instock	99.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
752	4864	0	8.30	0	outofstock	9.99	15154	NaN	NaN	NaN	NaN	NaN	NaN	
798	5017	0	0.00	0	outofstock	4.34	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
821	4233	0	0.00	0	outofstock	10.33	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Entrée [84]:
```
#création d'un dataframe avec les taux positifs
fusion_pos = fusion[fusion['taux_marge'] > 0]
#Afficher le prix minimum de la colonne "taux_marge"
tmpos_min = fusion_pos['taux_marge'].min()
print("Le taux de mrage minimun est de :", tmpos_min)
#Afficher le prix maximum de la colonne "taux_marge"
tmpos_max = fusion_pos['taux_marge'].max()
print("Le taux de mrage maximun est de :", tmpos_max)
```

```
Le taux de mrage minimun est de : 55.39739027283511
Le taux de mrage maximun est de : 129.6949650863653
```

Entrée [85]:
```
#création d'un dataframe avec le taux de marge moyen par type de produit
moy_marge = fusion_pos.groupby('product_type')['taux_marge'].mean().reset_index()
moy_marge
#Affichage dans un graphique du taux de marge par type de produit
cam = px.pie(moy_marge, names='product_type', values='taux_marge', title='Taux de Marge Moyen par Type de Produit')
cam.show()
```



Etape 5.5 - Analyse des correlations entre les variables stock, sales et price

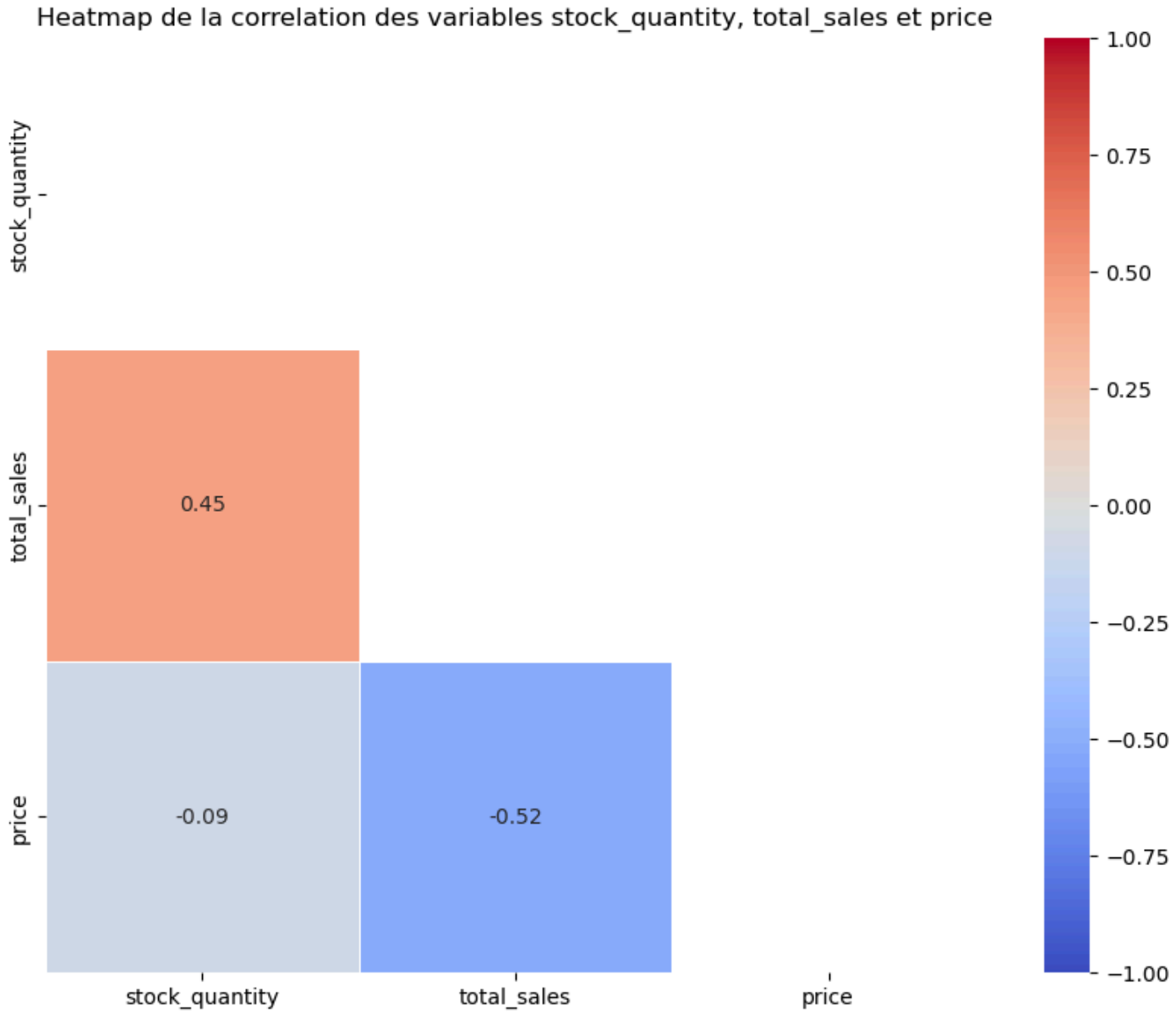
Entrée [86]:
```
fusion_pos.head(5)
```

Out[86]:

	product_id	onsale_web	price	stock_quantity	stock_status	purchase_price	id_web	virtual	downloadable	rating_count	average_rating	total_sales	tax_status	post
0	4337	1	83.0	145	instock	48.90	4679	0.0	0.0	0.0	0.0	0.0	taxable	
1	4350	1	79.5	145	instock	47.30	12588	0.0	0.0	0.0	0.0	7.0	taxable	
2	4334	1	49.0	142	instock	30.01	7818	0.0	0.0	0.0	0.0	7.0	taxable	
3	6126	1	135.0	138	instock	80.33	14923	0.0	0.0	0.0	0.0	5.0	taxable	
4	5025	1	112.0	136	instock	68.60	13914	0.0	0.0	0.0	0.0	6.0	taxable	

Entrée [87]:

```
############################
# Analyse des correlations #
############################

#Importation de Seaborn

#Création d'un heatmap de correlation avec les variables stock, sales et price
#on peut également créer un mask pour n'afficher qu'une demi heatmap
import seaborn as sns
correlation = fusion_pos[['stock_quantity','total_sales', 'price' ]].corr()
mask= np.triu(np.ones_like(correlation, dtype=bool))
plt.figure(figsize=(10,8))
sns.heatmap(correlation, annot=True, mask=mask, cmap='coolwarm', vmin=-1, vmax=1, linewidths=0.5)
plt.title('Heatmap de la correlation des variables stock_quantity, total_sales et price')
plt.show()
```


Heatmap de la correlation des variables stock_quantity, total_sales et price
stock_quantity
total_sales
price
0.45
-0.09
-0.52
stock_quantity
total_sales
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

Entrée [90]:
```
#Que peut-on conclure des correlations ?
print("stock_quantity et total_sales correlation de 0.27 cela indique une faible correlation positive")
print("stock_quantity et price correlation de -0.096 cela indique une tres faible correlation")
print("talal_sales et price correlation de -0.12 cela indique une faible correlation")
```

```
stock_quantity et total_sales correlation de 0.27 cela indique une faible correlation positive
stock_quantity et price correlation de -0.096 cela indique une tres faible correlation
talal_sales et price correlation de -0.12 cela indique une faible correlation
```

Etape 5.6 - Mettre à disposition la nouvelle table sur un fichier Excel

Entrée [103]:
```
#Mettre le dataset df_merge sur un fichier Excel
#Cette étape peut-être utile pour partager le résultat du dataset obtenu pour le partager avec les équipes.
chemin = r'C:\Users\maxen\OneDrive\Bureau\OpenClassRooms\fusion_pos.xlsx'
fusion_pos.to_excel(chemin, index=False)
```