TP OST Business Intelligence

NGUYEN Hoang Long
TONG Ngoc Chung

Exercice 1 : Nettoyage des données:

Le fichier origin :

```
        InvoiceNo StockCode  ... CustomerID         Country
0          536365    85123A  ...    17850.0  United Kingdom
1          536365     71053  ...    17850.0  United Kingdom
2          536365    84406B  ...    17850.0  United Kingdom
3          536365    84029G  ...    17850.0  United Kingdom
4          536365    84029E  ...    17850.0  United Kingdom
...           ...       ...  ...        ...             ...
541904     581587     22613  ...    12680.0          France
541905     581587     22899  ...    12680.0          France
541906     581587     23254  ...    12680.0          France
541907     581587     23255  ...    12680.0          France
541908     581587     22138  ...    12680.0          France

[541909 rows x 8 columns]
```

Le code:

```
#drop null
df = df.dropna()
print(df)
#Pas de valeurs négatives pour les quantités ou les prix
df = df[(df['Quantity']>0) & (df['UnitPrice']>0) ]
print(df)
#Le numéro de la facture est un nombre

df=df[pd.to_numeric(df['InvoiceNo'], errors='coerce').notnull()]
print(df)
```

Le fichier après nettoyer:

```
        InvoiceNo StockCode  ... CustomerID         Country
0          536365    85123A  ...    17850.0  United Kingdom
1          536365     71053  ...    17850.0  United Kingdom
2          536365    84406B  ...    17850.0  United Kingdom
3          536365    84029G  ...    17850.0  United Kingdom
4          536365    84029E  ...    17850.0  United Kingdom
...           ...       ...  ...        ...             ...
541904     581587     22613  ...    12680.0          France
541905     581587     22899  ...    12680.0          France
541906     581587     23254  ...    12680.0          France
541907     581587     23255  ...    12680.0          France
541908     581587     22138  ...    12680.0          France

[397884 rows x 8 columns]
```

Exercice 2 : L'agrégation et la définition de KPIs :

a, Le pouvoir d'achat des clients en moyenne par mois:

Le code:

```
df.InvoiceDate= pd.to_datetime(df.InvoiceDate)
print(np.dtype(df.InvoiceDate))

df['year_month'] = df['InvoiceDate'].dt.strftime('%Y-%m')
print(df)

df.CustomerID =  df.CustomerID.astype(int)
distinc_ID = df.CustomerID.nunique()
print(distinc_ID)

df["price_total"] = df.UnitPrice * df.Quantity
print(df)

price_total = df.groupby(["year_month"]).price_total.sum().reset_index(name='price_total')
nb_client =  df.groupby(["year_month"]).CustomerID.nunique().reset_index(name='nb_client')

KPI = pd.merge(price_total,nb_client, how ='left', on = ['year_month'])
KPI["KPIs"] = KPI.price_total / KPI.nb_client

print(KPI)
```

Après exécuter le code:

	year_month	price_total	nb_client	KPIs
0	2010-12	572713.890	885	647.134339
1	2011-01	569445.040	741	768.481835
2	2011-02	447137.350	758	589.890963
3	2011-03	595500.760	974	611.397084
4	2011-04	469200.361	856	548.131263
5	2011-05	678594.560	1056	642.608485
6	2011-06	661213.690	991	667.218658
7	2011-07	600091.011	949	632.340370
8	2011-08	645343.900	935	690.207380
9	2011-09	952838.382	1266	752.636953
10	2011-10	1039318.790	1364	761.963922
11	2011-11	1161817.380	1664	698.207560
12	2011-12	518192.790	615	842.589902

b, Est-ce qu'il y a des mois ou les ventes sont plus élevées que d'autres mois Indépendamment du nombre de clients ?

On peut voir qu'il y a 3 mois 9,10 et 11 de l'année 2011 ou les ventes sont plus élevés que d'autres mois (les autres mois sont environ 500000 - 600000 alors que ces 3 mois sont plus de 900000)

```
print(KPI.loc[KPI.price_total == KPI.price_total.max()])

n_largest = KPI.nlargest(3, 'price_total')
print(n_largest)
```

```
   year_month  price_total  nb_client      KPIs
11    2011-11   1161817.38       1664  698.20756
   year_month  price_total  nb_client       KPIs
11    2011-11  1161817.380       1664  698.207560
10    2011-10  1039318.790       1364  761.963922
9     2011-09   952838.382       1266  752.636953
```

c, Nous voulons exploiter les habitudes de dépenses pour ceux qui dépensent le plus (les 5 clients qui ont dépensé le plus tout au long de l'année):
- Combien de commandes en moyenne (vous devez penser aux nombres de factures par client) ont ces clients ?
- Est-ce qu'ils dépensent approximativement la même somme pour toutes leurs commandes ?

Les commandes en moyenne:

```
nb_facture_df = df.groupby(["CustomerID"]).InvoiceNo.nunique().reset_index(name='nb_facture')
nb_facture_df=nb_facture_df.sort_values(by = ['nb_facture'], ascending=True)
print(nb_facture_df)

#Combien de commandes en moyenne ?
print("moyenne de commande : " , nb_facture_df.nb_facture.mean())
```

```
      CustomerID  nb_facture
0          12346           1
1519       14420           1
3278       16812           1
3284       16820           1
3285       16823           1
...          ...         ...
1661       14606          93
562        13089          97
4010       17841         124
1879       14911         201
326        12748         209

[4338 rows x 2 columns]
moyenne de commande :  4.272014753342554
```

Il y a 4 facture en moyenne

Analyse des dépenses des clients:

```python
somme_depense_df =  df.groupby(["CustomerID", "InvoiceNo"]).price_total.sum().reset_index(name='somme')
print(somme_depense_df)
print("_____________________________________")

approximation = somme_depense_df.groupby(["CustomerID"]).somme.std().reset_index(name='stand_dev')
approximation = approximation.dropna() # drop les Nan values
approximation = approximation.sort_values(by = ['stand_dev'], ascending=True) # sort values ascending

print(approximation)
print(approximation.loc[approximation.stand_dev == 0])
print("_____________________________________")

for i in approximation.loc[approximation.stand_dev == 0, "CustomerID"]:
  print(somme_depense_df.loc[somme_depense_df.CustomerID == i])
```

Les dépenses des clients par facture:

```
       CustomerID InvoiceNo     somme
0           12346    541431  77183.60
1           12347    537626    711.79
2           12347    542237    475.39
3           12347    549222    636.25
4           12347    556201    382.52
...           ...       ...       ...
18527       18283    579673    223.61
18528       18283    580872    208.00
18529       18287    554065    765.28
18530       18287    570715   1001.32
18531       18287    573167     70.68

[18532 rows x 3 columns]
```

Après avoir la somme de dépense de chaque client, on utilise l'écart-type pour calculer l' approximation de la somme de dépense.

	CustomerID	stand_dev
1929	14987	0.000000
3540	17186	0.000000
3430	17029	0.000000
3653	17353	0.000000
3241	16766	0.000000
...	...	...
196	12590	6235.451444
2502	15749	6854.669639
55	12415	7774.349436
2011	15098	22226.807784
3008	16446	119123.945974

[2845 rows x 2 columns]

On peut voir qu'il y a des clients ont l'écart_type égal 0, ce sont des clients ont seulement 2 factures.

Exercice 3 : L'utilisation d'algorithme de fouilles de données pour établir de nouvelles corrélations et analyses :

a, Expliquer en quelques étapes c'est quoi les règles d'associations et pourquoi il est intéressant pour nos données d'utiliser cet algorithme ?

Règles d'association: est utilisé pour trouver relations entre les attributs dans de grandes bases de données. Par exemple, avec une règle d'association comme suivant:

Bread=> butter [support=2%, confidence-60%]

Cela signifie qu'il y a une transaction de 2% qui a acheté du pain et du beurre ensemble et que 60% des clients ont acheté du pain ainsi que du beurre.

Avec:

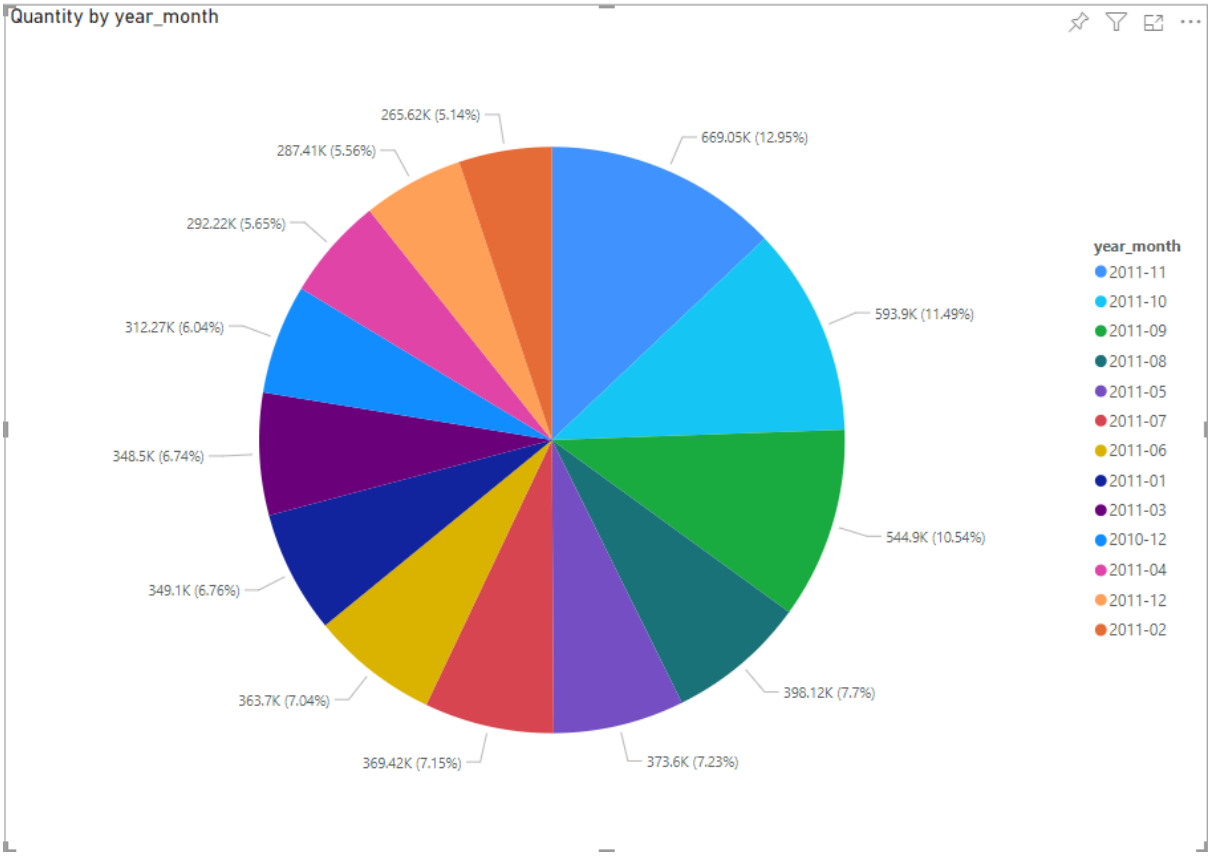
$$\text{Support (A)} = \frac{\text{Number of transaction in which A appears}}{\text{Total number of transactions}}$$

$$\text{Confidence (A}\longrightarrow\text{B)} = \frac{\text{Support(AUB)}}{\text{Support(A)}}$$

L'algorithme Apriori utilise deux étapes «joindre» et «élaguer» pour réduire l'espace de recherche. Donc il convient donc de réduire la vitesse d'analyse des grandes bases de données comme celles que nous utilisons. Ensuite, on peut trouver les règles d'association, donc on peut prédire les tendances d'achat de nos clients afin de pouvoir distribuer plus raisonnablement différents types d'articles.

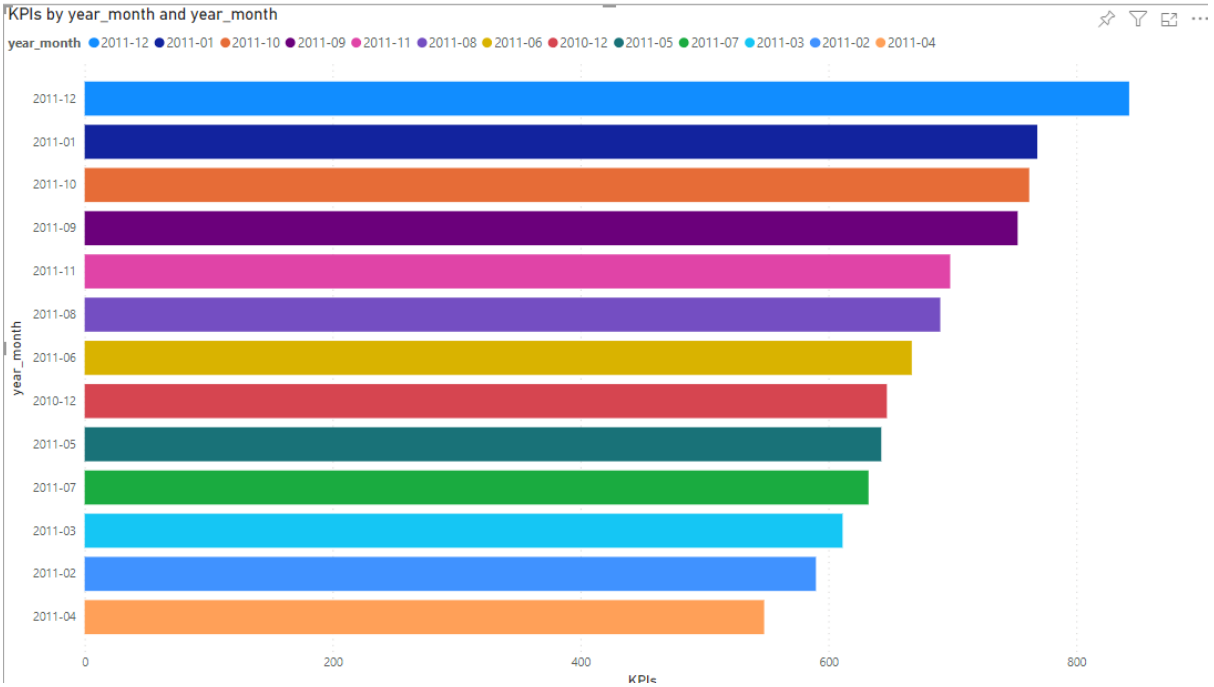
Visualisation des analyses:

Le tableau suivant montre le nombre total de ventes par mois



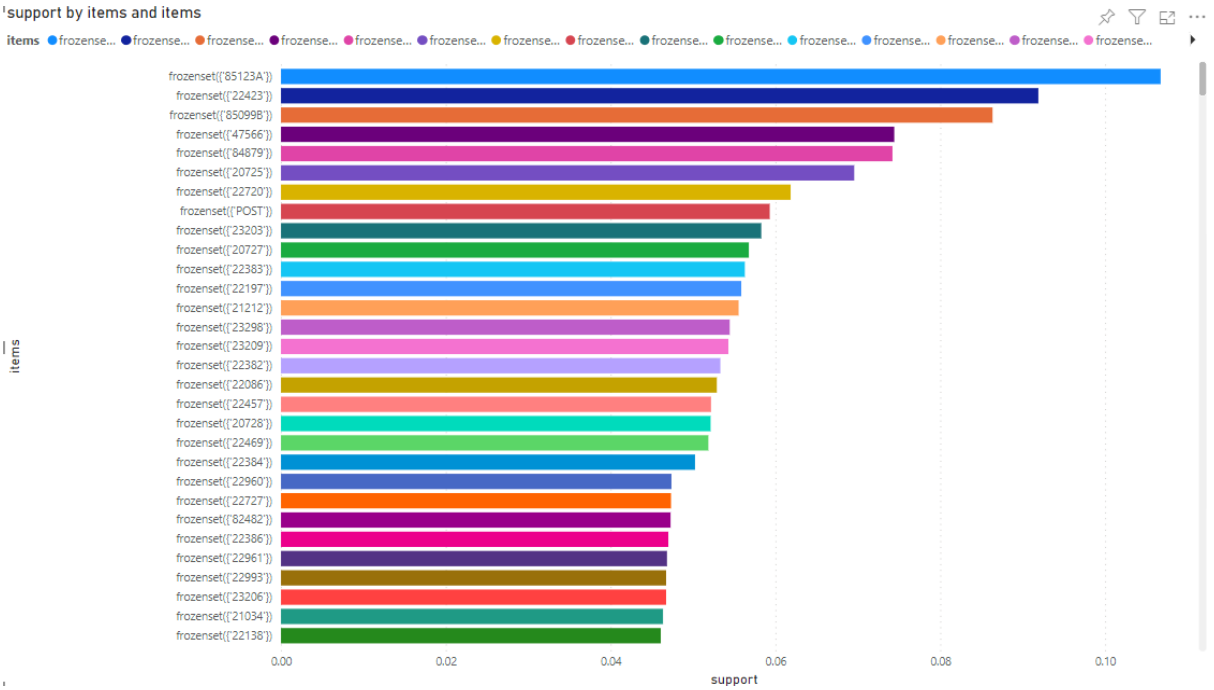
On peut voir que les mois 11/2011, 10/2011, 09/2011 sont les mois où la dépense en total sont plus élevée

Le tableau suivant montre le KPIs par mois



On peut voir que en 12/2011, la dépense de chaque client est la plus élevée.

Le tableau suivant montre la valeur support par items



Selon le diagramme, on peut voir que le items 85123A est l' item le plus être acheté .