Visualisation

December 1, 2024

1 Visualisation des données

[ ]:
```
import pandas as pd
```

[ ]:
```
df = pd.read_csv("sales_train.csv")
```

[ ]:
```
df
```

[ ]:
```
              date  date_block_num  shop_id  item_id  item_price  \
0       02.01.2013               0       59    22154      999.00
1       03.01.2013               0       25     2552      899.00
2       05.01.2013               0       25     2552      899.00
3       06.01.2013               0       25     2554     1709.05
4       15.01.2013               0       25     2555     1099.00
…              …               …      …       …          …
2935844 10.10.2015              33       25     7409      299.00
2935845 09.10.2015              33       25     7460      299.00
2935846 14.10.2015              33       25     7459      349.00
2935847 22.10.2015              33       25     7440      299.00
2935848 03.10.2015              33       25     7460      299.00

         item_cnt_day
0                 1.0
1                 1.0
2                -1.0
3                 1.0
4                 1.0
…                …
2935844           1.0
2935845           1.0
2935846           1.0
2935847           1.0
2935848           1.0

[2935849 rows x 6 columns]
```

Nous chargeons notre dataset afin de visualiser les données et en extraire des informations

[ ]:
```
# Filtrer les lignes avec item_price <= 0
negative_or_zero_price = df[df['item_price'] <= 0]

# Compter le nombre de lignes
count = negative_or_zero_price.shape[0]

print(f"Nombre de lignes où item_price est 0 ou négatif : {count}")
```

```
Nombre de lignes où item_price est 0 ou négatif : 1
```

[ ]:
```
# Supprimer les lignes où item_price <= 0
df = df[df['item_price'] > 0]
```

Nous avons vérifier s'il n'y a des valeurs de item_price <=0 et supprimer les lignes correspondantes, car on ne peut pas avoir le prix d'un article 0 ou négatif

[ ]:
```
# Filtrer les lignes avec item_price < 0
negative_or_zero_price = df[df['item_cnt_day'] < 0]

# Compter le nombre de lignes
count = negative_or_zero_price.shape[0]

print(f"Nombre de lignes où item_cnt_day est 0 ou négatif : {count}")
```

```
Nombre de lignes où item_cnt_day est 0 ou négatif : 7356
```

[ ]:
```
# Supprimer les lignes où item_cnt_day < 0
df = df[df['item_cnt_day'] >= 0]
```

Nous avons vérifier s'il n'y a des valeurs de item_cnt_day < 0 et supprimer les lignes correspondantes, car on peut pas avoir nombre d'article vendu < 0, ça peut etre des retour mais nous avons décider de les enlever et ne pas les traiter comme on a aucune informations sur ça

[ ]:
```
print(df.shape)
```

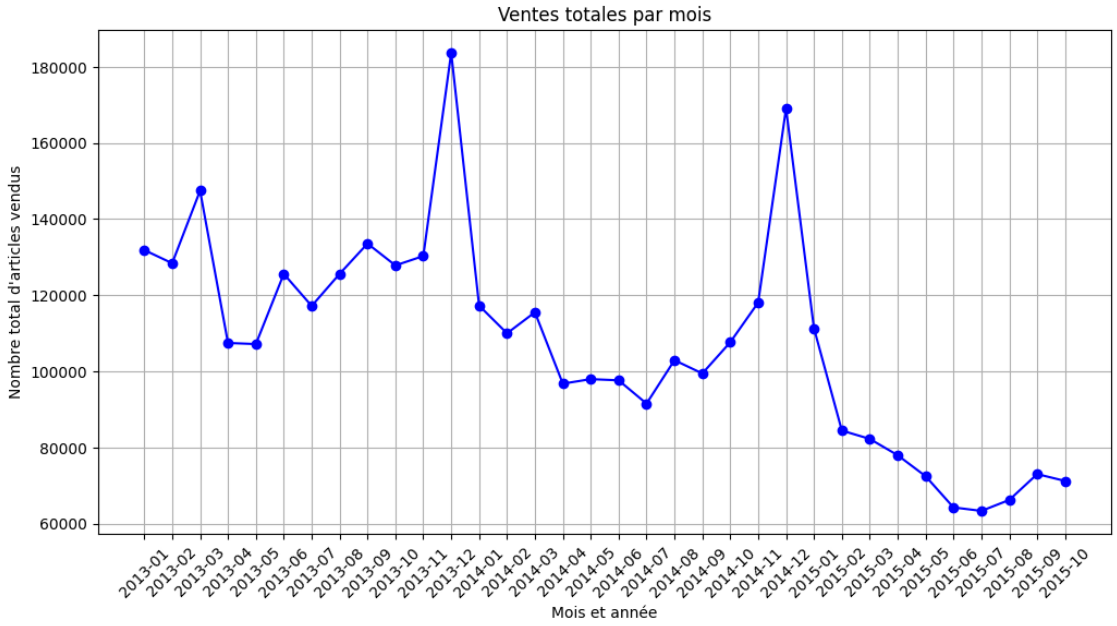
```
(2928492, 6)
```

Le nombre total des articles vendus par mois

[ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Assurez-vous que la colonne 'date' est bien en format datetime
df['date'] = pd.to_datetime(df['date'], format='%d.%m.%Y')

# Créez une colonne 'month_year' pour extraire le mois et l'année (ex : 'Janvier 2013')
df['month_year'] = df['date'].dt.to_period('M')
```

```
# Calcul de la vente mensuelle en agrégeant 'item_cnt_day' pour chaque mois␣
  ↪(par 'month_year')
monthly_sales = df.groupby('month_year')['item_cnt_day'].sum().reset_index()
```

[ ]:
```
# Visualisation
plt.figure(figsize=(12, 6))
plt.plot(monthly_sales['month_year'].astype(str),␣
  ↪monthly_sales['item_cnt_day'], color='blue', marker='o')
plt.title('Ventes totales par mois')
plt.xlabel('Mois et année')
plt.ylabel('Nombre total d\'articles vendus')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

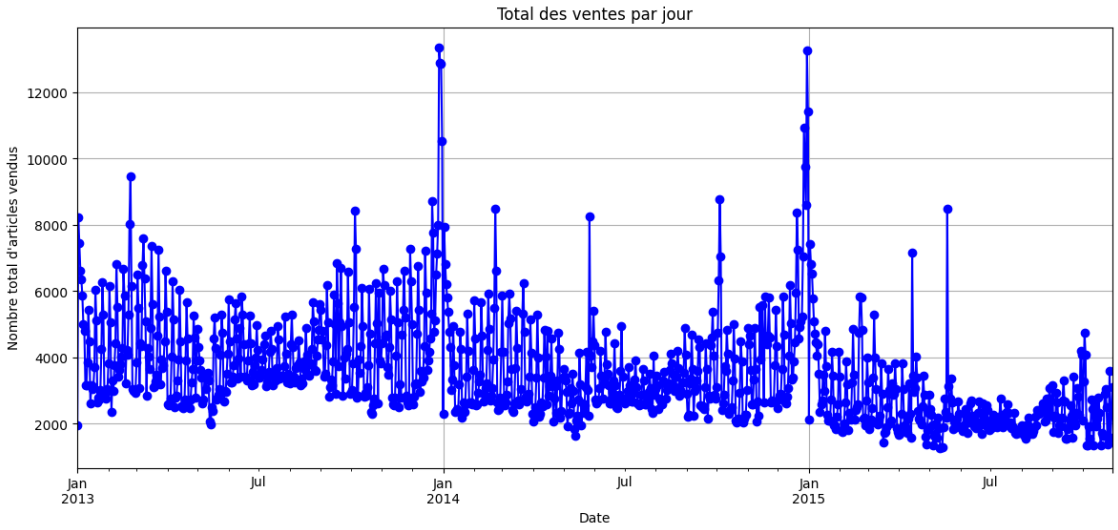


Nous avons affiché le nombre d'articles vendus par mois et avons observé un pic de ventes en novembre pour les années 2013 et 2014, suivi d'une baisse significative en 2015.

Nombre de vente par jour

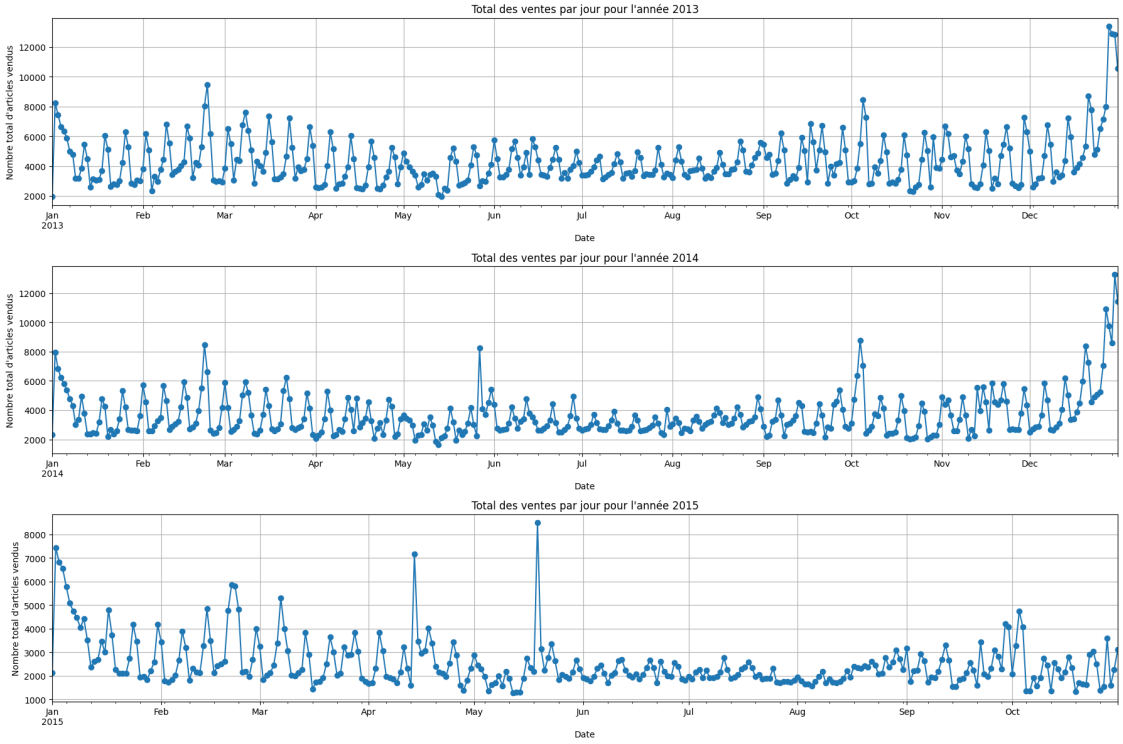
[ ]:
```
ventes_par_jour = df.groupby('date')['item_cnt_day'].sum()

plt.figure(figsize=(14, 6))
ventes_par_jour.plot(kind='line', color='blue', marker='o')
plt.title('Total des ventes par jour')
plt.xlabel('Date')
plt.ylabel('Nombre total d\'articles vendus')
```

```
plt.grid()
plt.show()
```



Nous avons affiché le nombre de ventes par jour, mais les résultats étaient difficiles à interpréter. Nous allons donc diviser cet affichage pour montrer le nombre de ventes par jour, séparé par les trois années.

[ ]:
```
df['date'] = pd.to_datetime(df['date'], format='%d.%m.%Y')
df['year'] = df['date'].dt.year

# Obtenir la liste des années uniques
years = sorted(df['year'].unique())

# Créer un graphique pour chaque année
plt.figure(figsize=(18, 12))

for i, year in enumerate(years):
    ventes_par_jour_annee = df[df['year'] == year].
 ↪groupby('date')['item_cnt_day'].sum()
    plt.subplot(len(years), 1, i + 1)
    ventes_par_jour_annee.plot(kind='line', marker='o')
    plt.title(f'Total des ventes par jour pour l\'année {year}')
    plt.xlabel('Date')
    plt.ylabel('Nombre total d\'articles vendus')
    plt.grid()

plt.tight_layout()
plt.show()
```

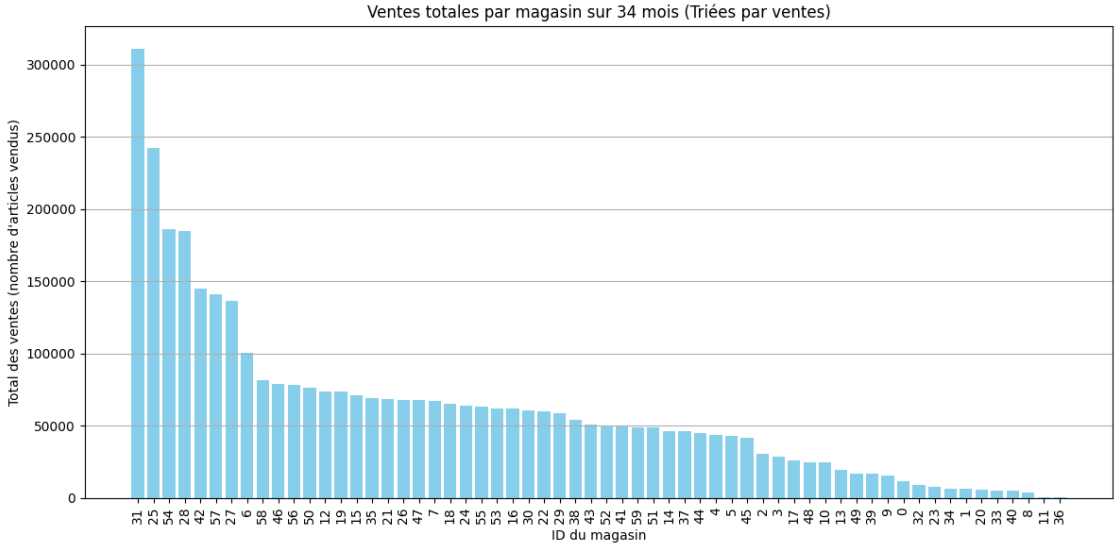


Maintenant que nous avons séparé les données sur trois ans, nous pouvons mieux visualiser les tendances. Nous observons que les ventes de janvier suivent un comportement similaire chaque année, avec une augmentation. De même, les ventes augmentent à la fin décembre. Cependant, nous remarquons qu'en mi-avril 2015, les ventes connaissent une forte hausse, contrairement aux années 2013 et 2014 où il n'y a pas de changement notable, et un phénomène similaire se produit à la mi-mai 2015.

Le nombre total des articles vendus par magasin

[ ]:
```
# Assurez-vous que la colonne 'date' est bien en format datetime
df['date'] = pd.to_datetime(df['date'], format='%d.%m.%Y')

# Créez une colonne 'month_year' pour extraire le mois et l'année
df['month_year'] = df['date'].dt.to_period('M')

# Calculez le total des ventes (item_cnt_day) pour chaque magasin sur la␣
 ↪période de 34 mois
total_sales_by_shop = df.groupby('shop_id')['item_cnt_day'].sum().reset_index()

# Trier les magasins par total des ventes, du plus élevé au plus bas
total_sales_by_shop_sorted = total_sales_by_shop.sort_values(by='item_cnt_day',␣
 ↪ascending=False)
```

```
# Visualisation sous forme de graphique
plt.figure(figsize=(12, 6))
plt.bar(total_sales_by_shop_sorted['shop_id'].astype(str),␣
 ↪total_sales_by_shop_sorted['item_cnt_day'], color='skyblue')

# Titre et labels
plt.title('Ventes totales par magasin sur 34 mois (Triées par ventes)')
plt.xlabel('ID du magasin')
plt.ylabel('Total des ventes (nombre d\'articles vendus)')

# Rotation des labels sur l'axe X pour éviter les chevauchements
plt.xticks(rotation=90)

# Ajouter une grille sur l'axe Y
plt.grid(True, axis='y')

# Afficher le graphique
plt.tight_layout()  # Pour ajuster le graphique en cas de labels longs
plt.show()
```

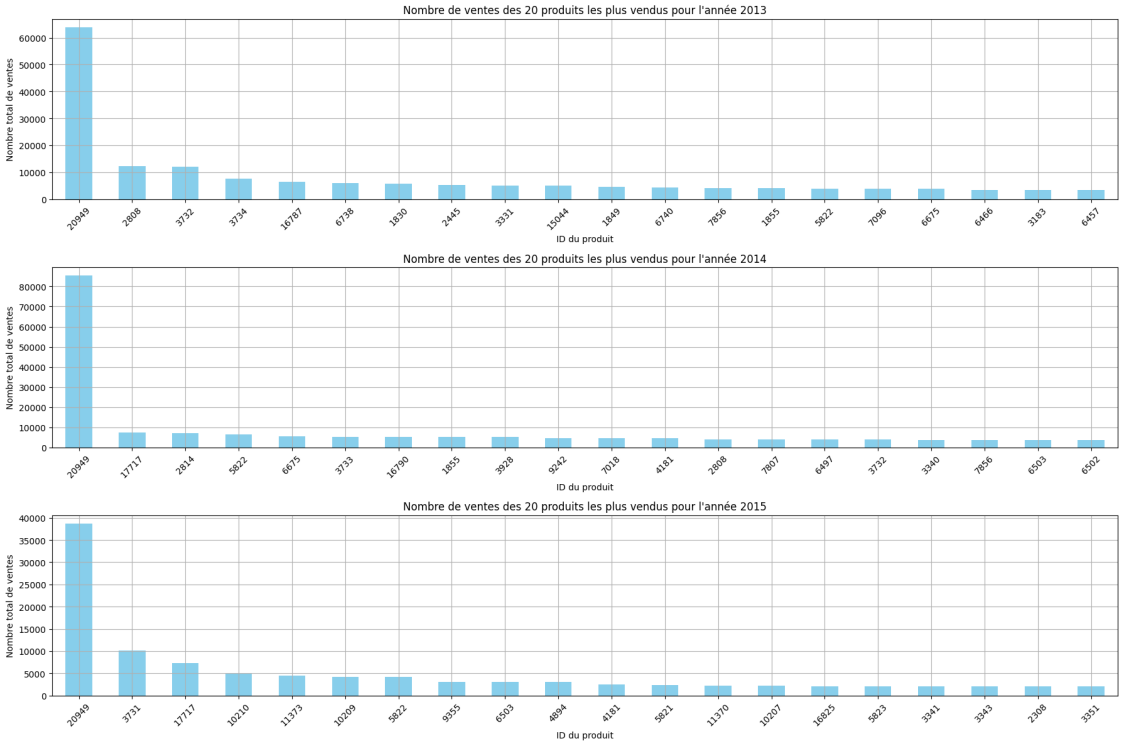


Nous avons affiché le nombre de ventes par magasin et constaté que le magasin 31 enregistre les ventes les plus élevées, tandis que le magasin 36 affiche des ventes très faibles.

Nombre de vente par article

[ ]:
```
plt.figure(figsize=(18, 12))

for i, year in enumerate(years):
```

```
    ventes_par_produit_annee = df[df['year'] == year].
 ↪groupby('item_id')['item_cnt_day'].sum()
    top_ventes_par_produit = ventes_par_produit_annee.nlargest(20)  # Afficher␣
 ↪les 20 produits les plus vendus
    plt.subplot(len(years), 1, i + 1)
    top_ventes_par_produit.plot(kind='bar', color='skyblue')
    plt.title(f'Nombre de ventes des 20 produits les plus vendus pour l\'année␣
 ↪{year}')
    plt.xlabel('ID du produit')
    plt.ylabel('Nombre total de ventes')
    plt.xticks(rotation=45)
    plt.grid()

# Ajuster l'espacement entre les graphiques pour plus de lisibilité
plt.subplots_adjust(hspace=1.0)  # Augmenter la valeur pour plus d'espace entre␣
 ↪les graphiques

plt.tight_layout()
plt.show()
```



Les graphiques montrent une forte concentration des ventes sur un produit dominant chaque année, avec une légère variation d'une année à l'autre. En 2013, le produit 20949 domine largement les ventes, suivi par quelques produits secondaires, et cette tendance se renforce en 2014 avec une

augmentation des ventes du produit phare, tandis que les autres produits restent à des niveaux beaucoup plus bas. Cependant, en 2015, bien que le produit 20949 reste en tête, ses ventes diminuent, et les autres produits affichent des performances plus équilibrées. Cela suggère une diversification des préférences des consommateurs, avec une diminution de la domination d'un seul produit. Les recommandations incluent la diversification des produits pour réduire la dépendance au produit principal et une analyse approfondie des causes de la baisse des ventes en 2015.

Chiffre d'affaire par magasin

[ ]:
```
df['chiffre_affaire'] = df['item_price'] * df['item_cnt_day']  # Calcul du␣
 ↪chiffre d'affaires
plt.figure(figsize=(18, 12))

for i, year in enumerate(years):
    chiffre_affaire_par_magasin_annee = df[df['year'] == year].
 ↪groupby('shop_id')['chiffre_affaire'].sum()
    plt.subplot(len(years), 1, i + 1)
    chiffre_affaire_par_magasin_annee.plot(kind='bar', color='orange')
    plt.title(f'Chiffre d\'affaires par magasin pour l\'année {year}')
    plt.xlabel('ID du magasin')
    plt.ylabel('Chiffre d\'affaires total')
    plt.xticks(rotation=45)
    plt.grid()

# Ajuster l'espacement entre les graphiques pour plus de lisibilité
plt.subplots_adjust(hspace=1.0)

plt.tight_layout()
plt.show()
```

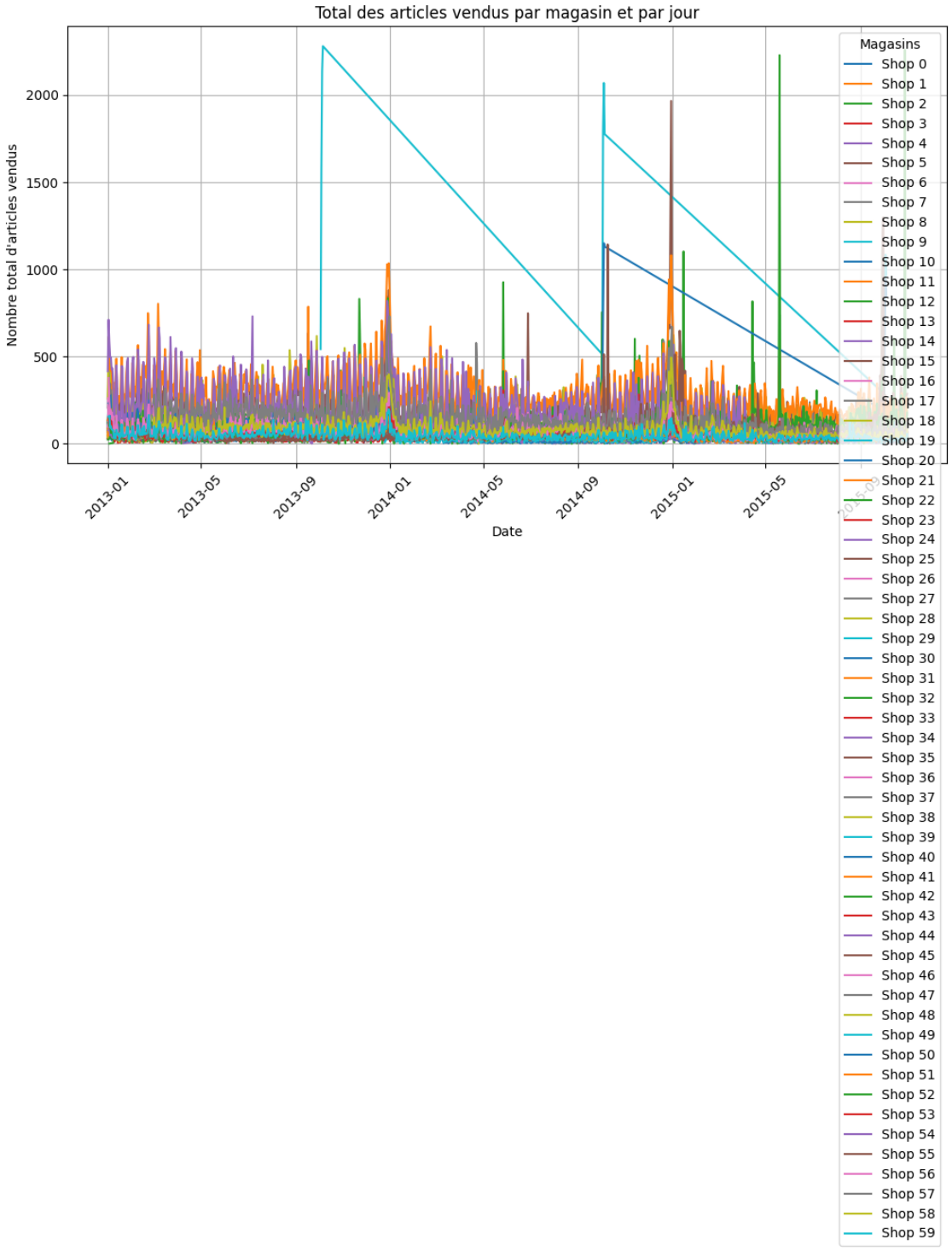
Les graphiques montrent le chiffre d'affaires total par magasin sur une période de trois ans, avec une répartition inégale où certains magasins génèrent des revenus nettement plus élevés que d'autres. Les magasins ID 31, 54, et 25 sont les principaux contributeurs sur les trois années. En 2013, le magasin 31 est le leader incontesté, suivi de près par les magasins 54 et 25, tandis que d'autres magasins génèrent des chiffres bien inférieurs. En 2014, bien que le magasin 31 reste en tête, certains magasins secondaires, comme les ID 10 et 42, enregistrent une progression notable. En 2015, le magasin 31 demeure dominant, mais son chiffre d'affaires diminue légèrement, et le magasin 25 dépasse le magasin 54, signalant une redistribution des revenus. La tendance montre une diversification des performances, avec une répartition plus équilibrée des revenus parmi les magasins, ce qui pourrait refléter une saturation des marchés des magasins dominants et une amélioration des autres.

Total des articles par magasin et par jour

[ ]:
```
# Assurez-vous que la colonne 'date' est bien en format datetime
df['date'] = pd.to_datetime(df['date'], format='%d.%m.%Y')

# Calculez le total des articles vendus par magasin et par jour
total_sales_by_shop_day = df.groupby(['shop_id', 'date'])['item_cnt_day'].sum().
  ↪reset_index()


# Visualisation sous forme de graphique
plt.figure(figsize=(12, 6))
```

```
# Pour chaque magasin, on trace une courbe montrant le total des ventes sur la␣
 ↪période
for shop in total_sales_by_shop_day['shop_id'].unique():
    shop_sales = total_sales_by_shop_day[total_sales_by_shop_day['shop_id'] ==␣
 ↪shop]
    plt.plot(shop_sales['date'], shop_sales['item_cnt_day'], label=f'Shop␣
 ↪{shop}')

# Ajouter les détails du graphique
plt.title('Total des articles vendus par magasin et par jour')
plt.xlabel('Date')
plt.ylabel('Nombre total d\'articles vendus')
plt.legend(title="Magasins")
plt.grid(True)

# Afficher le graphique
plt.xticks(rotation=45)  # Pour améliorer la lisibilité des dates
plt.tight_layout()  # Ajuster l'affichage pour éviter les chevauchements
plt.show()
```

```
<ipython-input-18-4588f8600c77>:25: UserWarning: Tight layout not applied. The
bottom and top margins cannot be made large enough to accommodate all axes
decorations.
  plt.tight_layout()  # Ajuster l'affichage pour éviter les chevauchements
```



Avec cette visualisation, nous avons observé l'évolution des ventes totales par magasin et par jour. Les courbes montrent une forte variabilité dans les ventes entre les différents magasins, avec certains magasins (comme le magasin 19) générant des ventes beaucoup plus importantes que d'autres. Cette concentration des ventes sur quelques magasins est particulièrement évidente, ce qui reflète

probablement une différence dans la localisation des magasins ou dans leur capacité à attirer des clients. De plus, les ventes semblent suivre des tendances saisonnières avec des pics évidents pendant certaines périodes de l'année, comme les mois de fin d'année(mois de novembre). Enfin, bien que certains magasins connaissent une croissance continue, d'autres présentent des périodes de stagnation ou de baisse, ce qui pourrait être lié à des changements dans les comportements d'achat affectant ces magasins spécifiques.

[ ]: