# Logements Airbnb dans les grandes villes européennes

Anastasiia Savelkova
Victoria Vidal
Moheb Nazeer
Théo Linale

**Master 1 Économétrie Statistiques**

Paris I Panthéon-Sorbonne

**Langage de programmation**

Enseignante : Loukah Imane

Novembre 2024

# Table des matières

# Introduction

Airbnb, en tant que plateforme mondiale de location de logements, joue un rôle majeur dans l'industrie du tourisme et de l'hébergement. Avec environ 5 649 835 visites par jour, elle offre aux voyageurs des alternatives diversifiées aux hôtels traditionnels.

L'objectif principal est de développer un moteur de recherche et de recommandation efficace pour les utilisateurs, en s'appuyant sur une analyse approfondie des données. Avec 18 variables différentes, il est possible de mieux saisir les dynamiques sociales, économiques et géographiques ainsi que leur impact sur les prix des logements et les préférences des utilisateurs. De plus, ce sujet présente un grand intérêt sur le plan de la programmation. L'utilisation de Python pour filtrer, manipuler et visualiser les données constitue le cœur de cette analyse. L'application de modèles de régression permet d'approfondir la compréhension des relations entre les différentes variables et d'améliorer la précision des recommandations. Ainsi, ce projet allie une analyse de données rigoureuse et une approche technique avancée pour valoriser les informations issues d'Airbnb.

Le rapport comprend la restitution des analyses de la première partie avec la visualisation et les statistiques descriptives des variables (I) ; la méthodologie, qui présente la description des étapes de développement, les bibliothèques utilisées et la répartition du travail (II) ; la conception : l'architecture du programme (III) ; l'implémentation avec les fonctionnalités développées (IV) et les méthodes de test appliquées au projet (V). Enfin, la conclusion aborde les perspectives du projet, suivie de la bibliographie des sources utilisées.

# I. Analyse des résultats obtenus dans la Partie 1

Cette partie est consacrée à l'analyse des caractéristiques influant sur la tarification des logements, avec la statistique descriptive des variables et la visualisation des relations entre elles. Cette étape était préalable pour comprendre les effets géographiques, sociaux et des évaluations par les utilisateurs sur les prix.

Tout au long de la section I. Analyse, nous avons mis l'accent sur les graphiques et leur interprétation plutôt que sur le code, qui n'était pas particulièrement complexe, notamment grâce aux ressources et aux plateformes dédiées à Python mentionnées dans la bibliographie.

## I.1 Etape prélable : Description des données

D'abord, on a effectué une description classique des données pour deux périodes grâce aux fonctions `.info()` et `.describe()`.

### a) Description du Dataset : Paris, *weekdays*

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3130 entries, 0 to 3129
Data columns (total 20 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Unnamed: 0                  3130 non-null   int64
 1   realSum                     3130 non-null   float64
 2   room_type                   3130 non-null   object
```

```
 3   room_shared                  3130 non-null   bool
 4   room_private                 3130 non-null   bool
 5   person_capacity              3130 non-null   float64
 6   host_is_superhost            3130 non-null   bool
 7   multi                        3130 non-null   int64
 8   biz                          3130 non-null   int64
 9   cleanliness_rating           3130 non-null   float64
 10  guest_satisfaction_overall   3130 non-null   float64
 11  bedrooms                     3130 non-null   int64
 12  dist                         3130 non-null   float64
 13  metro_dist                   3130 non-null   float64
 14  attr_index                   3130 non-null   float64
 15  attr_index_norm              3130 non-null   float64
 16  rest_index                   3130 non-null   float64
 17  rest_index_norm              3130 non-null   float64
 18  lng                          3130 non-null   float64
 19  lat                          3130 non-null   float64
dtypes: bool(3), float64(12), int64(4), object(1)
memory usage: 425.0+ KB
None
```

Les types de données sont variés : on observe des variables de type `boolean` pour indiquer des caractéristiques comme le type de chambre (`room_shared`, `room_private`, `host_is_superhost`), des variables `integer` pour les informations numériques (`multi` et `biz`) et des variables `float` pour les valeurs continues, telles que les prix (`realSum`), les évaluations (`cleanliness_rating`, `guest_satisfaction_overall`), ainsi que les distances et indices d'attraction. Une variable de type `object` est utilisée pour le type d'hébergement (`room_type`).

On a fait la statistique descriptive pour la variable `realSum` des logements à Paris, pour mieux comprendre la structure des données et faire une première visualisation.

### b) Analyse des prix, Paris

- **Prix moyen** :
  - Weekends : 387,03 €
  - Weekdays : 398,79 €
- **Écart-type** :
  - Weekends : 260,08 €
  - Weekdays : 396,37 €
- **Prix minimum** :
  - Weekends : 95,30 €
  - Weekdays : 92,74 €
- **Prix maximum** :
  - Weekends : 4 188,41 €
  - Weekdays : 16 445,61 €

On observe que les prix sont légèrement plus élevés en semaine. Cependant, l'écart-type est également plus grand, indiquant une variabilité plus importante des prix en semaine.

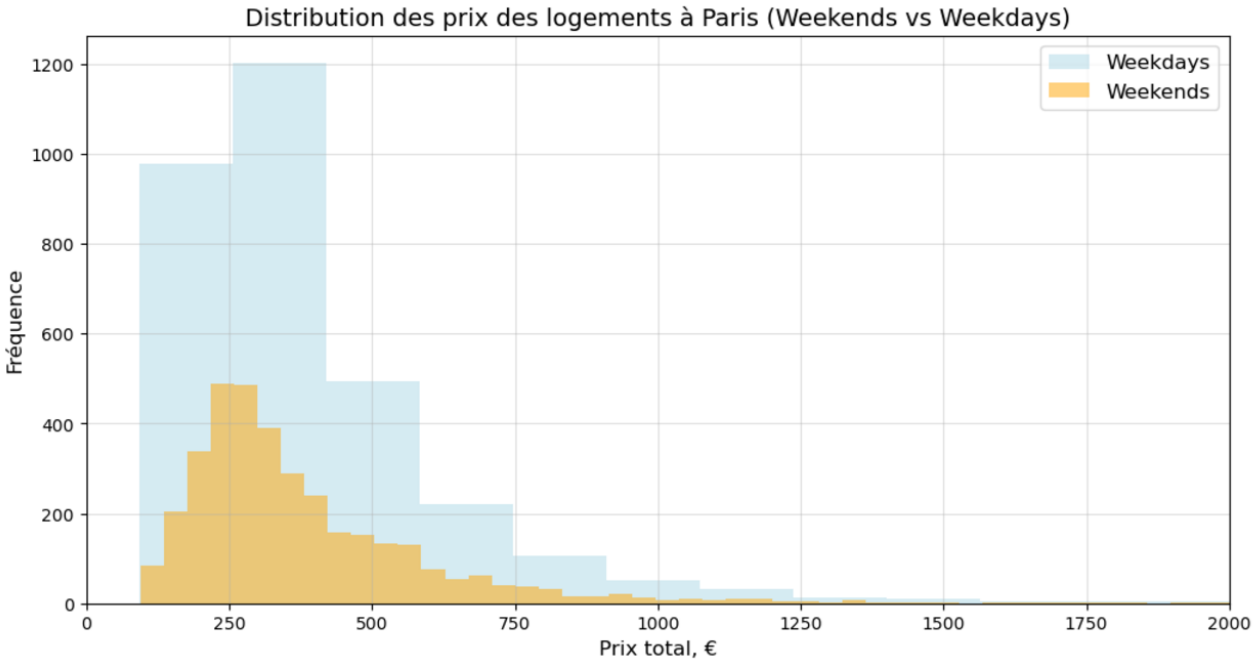


FIGURE 1 – Distribution des prix des logements à Paris

### c) Visualisation de la distribution des prix pour Paris

Les prix des weekends et weekdays ont des distributions assez similaires, mais on observe une plus grande variabilité des prix en semaine avec des valeurs extrêmes plus élevées.

## I.2 Question 1 : La différence de prix entre les logements en *weekdays* et *weekends* pour chaque ville

Pour cette question on a analysé les prix moyens des logements en deux périodes.

### a) Les prix moyens pour chaque ville

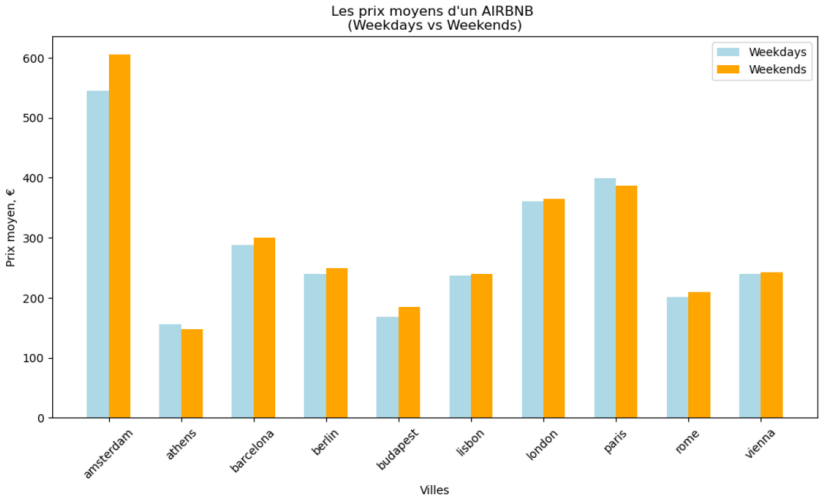


FIGURE 2 – Distribution des prix moyens d'un Airbnb (Weekdays vs Weekends)

La Figure 2 montre que, généralement, le prix pendant les *weekdays* est inférieur à celui des *weekends*, ce qui est logique étant donné qu'il y a plus de touristes durant les *weekends*. Cependant, pour des villes comme Athènes et Paris, cette tendance est inversée. On observe également que, pour Amsterdam, l'écart entre les prix des *weekdays* et des *weekends* est plus marqué et est d'environ 59 €.

### b) Description des prix moyens en : Villes, *Weekdays* et *Weekends*

| Ville | Weekdays | Weekends |
|---|---|---|
| amsterdam | 545.020526 | 604.828018 |
| athens | 155.866982 | 147.580456 |
| barcelona | 288.391667 | 300.277940 |
| berlin | 240.220422 | 249.252516 |
| budapest | 168.429367 | 185.120628 |
| lisbon | 236.345459 | 240.044051 |
| london | 360.230348 | 364.389747 |
| paris | 398.786678 | 387.028589 |
| rome | 201.618053 | 209.130063 |
| vienna | 240.384834 | 242.739524 |

Le tableau présente les prix moyens des Airbnb par ville, répartis entre les jours de semaine et les weekends. On observe une variation notable des prix selon la semaine, ce qui peut s'expliquer par la demande touristique différente pendant la semaine et le week-end.

## I.3 Question 2 : Les villes les plus chères et les moins chères avec les appartements privés pour au moins 4 personnes

### a) Filtrage des données

Pour cette question, avant de comparer les moyennes des prix et chercher le maximum et le minimum, on a d'abord filtré les données.

```
for ville, chemin_fichier in files_weekdays.items():
    df_weekdays = pd.read_csv(chemin_fichier)
    filtre = (df_weekdays["person_capacity"] >= 4) & (df_weekdays["
       room_private"] == True)
    df_filtre = df_weekdays[filtre]
    prix_moyen_weekdays[ville] = df_filtre["realSum"].mean()

for ville, chemin_fichier in files_weekends.items():
    df_weekends = pd.read_csv(chemin_fichier)
    filtre = (df_weekends["person_capacity"] >= 4) & (df_weekends["
       room_private"] == True)
    df_filtre = df_weekends[filtre]
    prix_moyen_weekends[ville] = df_filtre["realSum"].mean()
```

Après l'exécution du code, on a obtenu les résultats suivants.

***Weekdays*** :

— La ville la plus chère est **Amsterdam** avec un prix moyen de 567.62 €.

— La ville la moins chère est **Budapest** avec un prix moyen de 132.24 €.

***Weekends*** :

— La ville la plus chère est **Amsterdam** avec un prix moyen de 639.85 €.

— La ville la moins chère est **Budapest** avec un prix moyen de 158.69 €.

### b) Scatter plot pour les prix des appartements privés pour au moins 4 personnes

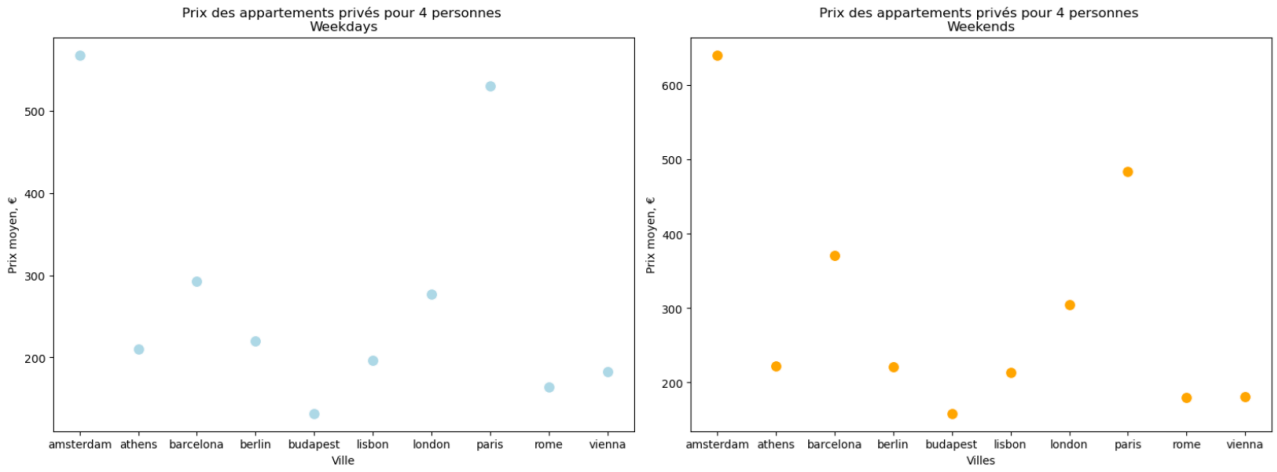


FIGURE 3 – Nuage des points (Weekdays vs Weekends)

La Figure 3 confirme l'output obtenu. On voit que pour les weekdays et les weekends, les points les plus hauts et les plus bas correspondent respectivement aux villes d'Amsterdam et de Budapest.

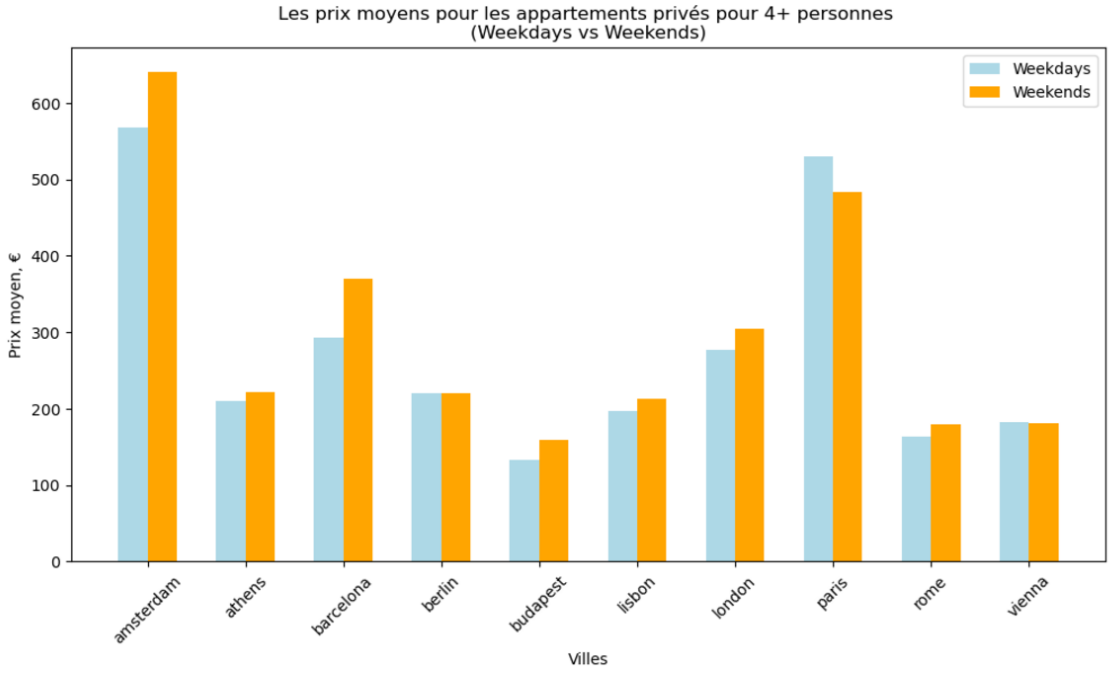


FIGURE 4 – Histogramme des prix moyens (Weekdays vs Weekends)

### c) Prix moyen global des appartements privés (4 personnes ou plus)

Pour analyser le prix moyen global des appartements privés, nous avons commencé par fusionner les données relatives aux jours de semaine (weekdays) et aux week-ends (weekends).
La fusion a été réalisée en calculant une moyenne générale pour chaque ville à l'aide de la formule suivante :

$$\text{Prix moyen global} = \frac{\text{Prix moyen (weekdays)} + \text{Prix moyen (weekends)}}{2}$$

Cette méthode nous permet d'obtenir une vue d'ensemble des prix pour toutes les villes étudiées, indépendamment du jour de la semaine. Le graphique ci-dessous illustre les résultats obtenus, avec une comparaison des prix moyens pour les appartements accueillant 4 personnes ou plus.

**Moyennes globales pour chaque ville :**

- La moyenne globale pour **Amsterdam** est de 603.74 €.
- La moyenne globale pour **Athens** est de 216.22 €.
- La moyenne globale pour **Barcelona** est de 331.76 €.
- La moyenne globale pour **Berlin** est de 220.35 €.
- La moyenne globale pour **Budapest** est de 145.46 €.
- La moyenne globale pour **Lisbon** est de 204.61 €.
- La moyenne globale pour **London** est de 290.87 €.
- La moyenne globale pour **Paris** est de 506.90 €.
- La moyenne globale pour **Rome** est de 171.86 €.
- La moyenne globale pour **Vienna** est de 182.20 €.

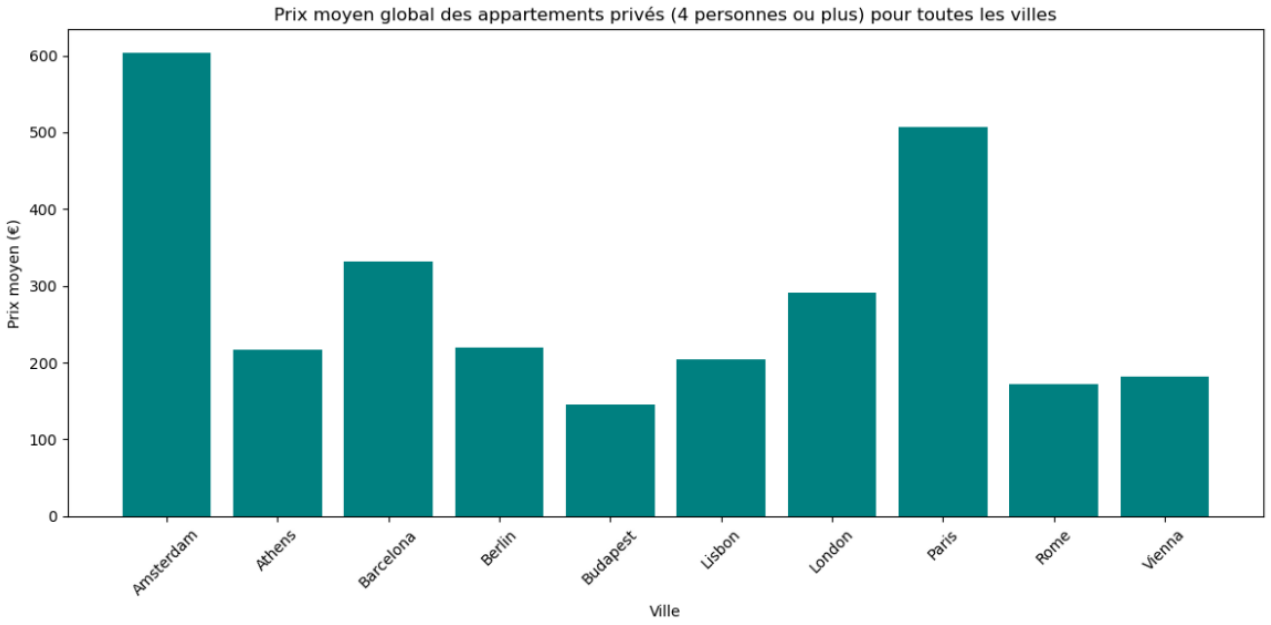


FIGURE 5 – Prix globale (Weekdays vs Weekends)

## I.4 Question 3 : L'impact des éléments geographiques sur la tarification des logements

D'abord, il faut définir les variables que l'on peut associer aux éléments géographiques :

- `dist` : Distance du centre-ville

- `metro_dist` : Distance de la station de métro la plus proche en km

Ainsi, l'indice d'attraction de l'emplacement du logement *attr_index* peut affecter la tarification. La variable de prix total de l'hébergement *realSum* correspond à la tarification et est de type `integer`. Toutes les variables sont de type `float`.

Il y a plusieurs manières d'explorer l'impact des éléments géographiques sur la tarification des logements. Voici quelques idées de notre groupe :

### a) Analyse des corrélations

Il est important d'examiner la corrélation entre les variables car cela permet de comprendre les relations sous-jacentes et d'identifier les facteurs influençant le prix de l'hébergement.

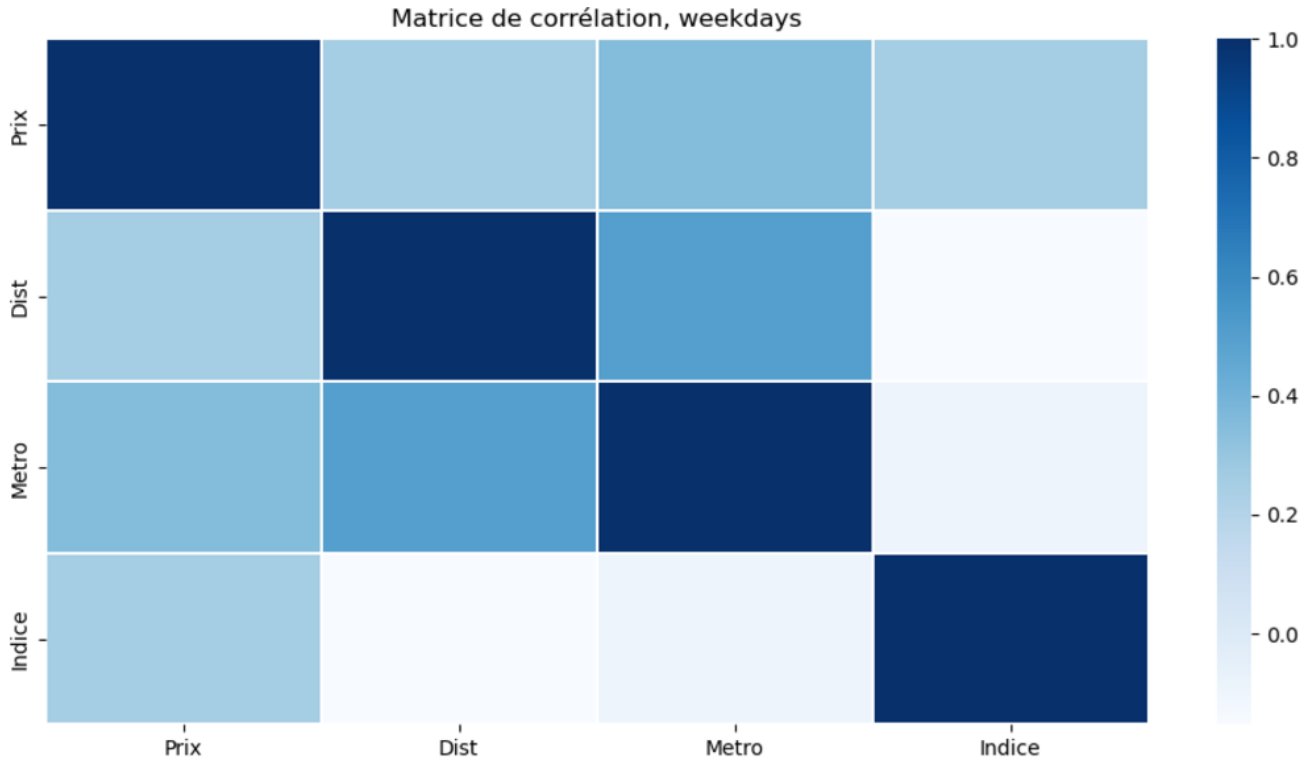


FIGURE 6 – Matrice de corrélation pour les weekdays

| Variables | Prix | Dist |
|---|---|---|
| Prix | 1.000000 | 0.254798 |
| Dist | 0.254798 | 1.000000 |

| Variables | Prix | Metro |
|---|---|---|
| Prix | 1.000000 | 0.354382 |
| Metro | 0.354382 | 1.000000 |

| Variables | Prix | Indice |
|---|---|---|
| Prix | 1.0000 | 0.2535 |
| Indice | 0.2535 | 1.0000 |

TABLE 1 – Corrélation entre les prix et les éléments géographiques

Il existe une corrélation positive entre la tarification des logements et les deux éléments géographiques : la distance du centre-ville (25.5%) et la distance de la station de métro la plus proche (35.4%). Cela implique qu'à mesure que la distance du centre-ville ou la distance à la station de métro augmente, le prix des logements a tendance à augmenter.

Cependant, ces relations sont faibles, ce qui suggère que la géographie n'est pas un facteur prédominant dans la détermination des prix des logements. Cependant, on s'attend généralement à ce que les prix des logements diminuent à mesure que la distance au centre-ville ou à une station de métro augmente. Mais il faut tenir compte que les logements situés à une plus grande distance du centre ou de métro peuvent, par exemple, se trouver dans des quartiers calmes et peuvent offrir plus d'espace, moins de bruit et de pollution. De plus, on analyse les grandes villes touristiques qui ont plusieurs pôles d'attractivité en dehors du centre comme les quartiers historiques ou les centres culturels éloignés du centre.

Ainsi, la corrélation entre le prix d'hébergement et l'indice d'attraction de l'emplacement est positive (25.35%) mais n'est pas très élevée aussi. Cela indique que les logements situés dans des zones avec un indice d'attraction élevé ont légèrement tendance à être plus chers.

De plus, la corrélation entre les prix moyens en *weekdays* et en *weekends* est de 99.3%, ce qui signifie que les variations de prix pour ces deux périodes suivent un comportement similaire. Les facteurs géographiques, les dynamiques sociales et les évaluations qui influencent les prix sont assez constants à la fois pendant la semaine et le weekend. Dans cette condition, on peut traiter les deux périodes comme étant assez homogènes, elles seront affectées de la même façon.

### b) Relation entre la tarification des logements et la distance du centre-ville

| City | Price at 0 km | Price at 5 km | Price at 10 km |
|---|---|---|---|
| Amsterdam | 704.55 | 333.72 | 306.53 |
| Athens | 203.65 | 115.83 | NaN |
| Barcelona | 305.92 | 289.09 | NaN |
| Berlin | 250.99 | 215.50 | 196.10 |
| Budapest | 194.49 | 140.22 | 113.21 |
| Lisbon | 269.75 | 225.90 | NaN |
| London | 835.15 | 311.78 | 176.31 |
| Paris | 508.06 | 398.23 | NaN |
| Rome | 182.44 | 157.62 | NaN |
| Vienna | 314.67 | 200.07 | 146.00 |

TABLE 2 – Tableau récapitulatif des prix en fonction de la distance au centre-ville

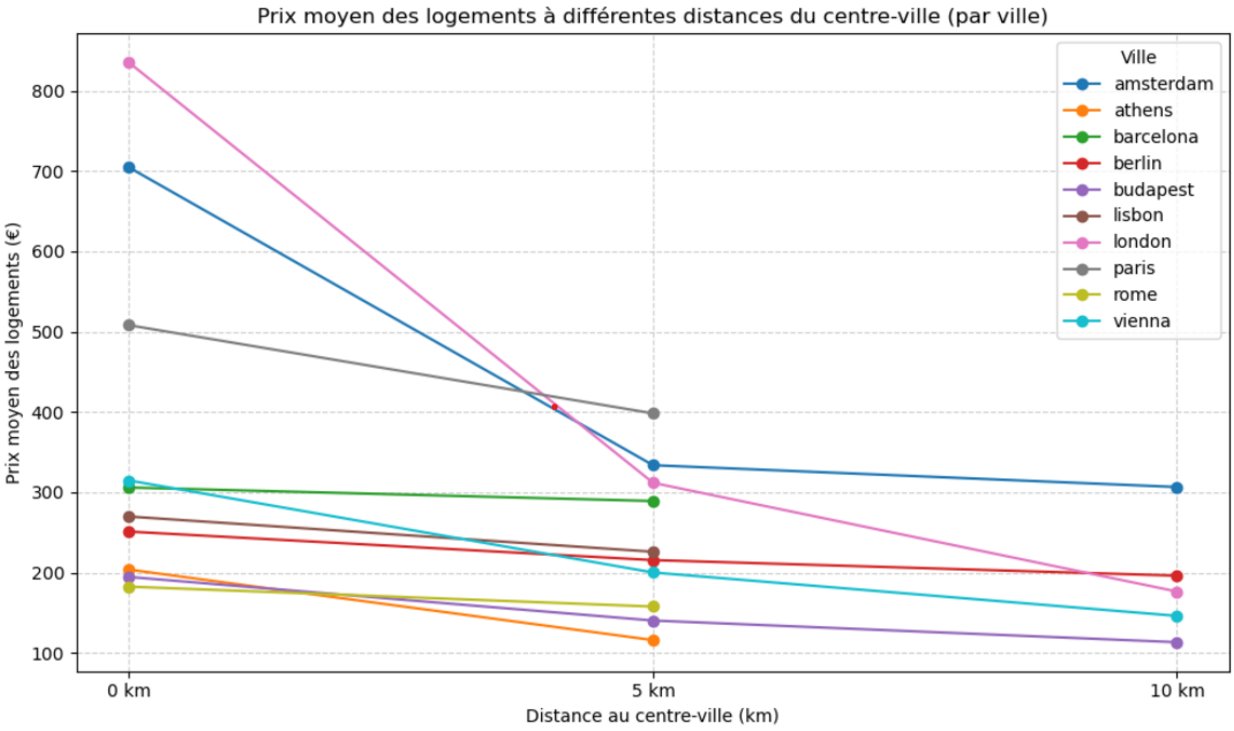


FIGURE 7 – Relation entre les prix et les distances du centre-ville

Les données montrent une forte variation des prix des logements en fonction de la distance au centre-ville. Les villes comme Amsterdam, avec un prix de 704,55 €dans le centre, et Londres, avec un prix de 835,15 €, affichent des valeurs très élevées, soulignant une forte demande pour les logements dans ces zones centrales. En revanche, des villes comme Athènes, où le prix est de 203,65 €, et Budapest, à 194,49 €, maintiennent des prix nettement plus accessibles, même dans le centre, reflétant un coût de vie plus bas ou une moindre pression sur l'immobilier.

Certaines villes, telles que Barcelone, avec un prix de 305,92 €, et Paris, à 508,06 €, montrent une relative stabilité des prix jusqu'à 5 km du centre, tandis que pour 10 km, des données sont manquantes, limitant l'analyse des zones plus éloignées. Ces résultats mettent en évidence des différences significatives entre les villes, largement influencées par les spécificités économiques et la structure urbaine locale.

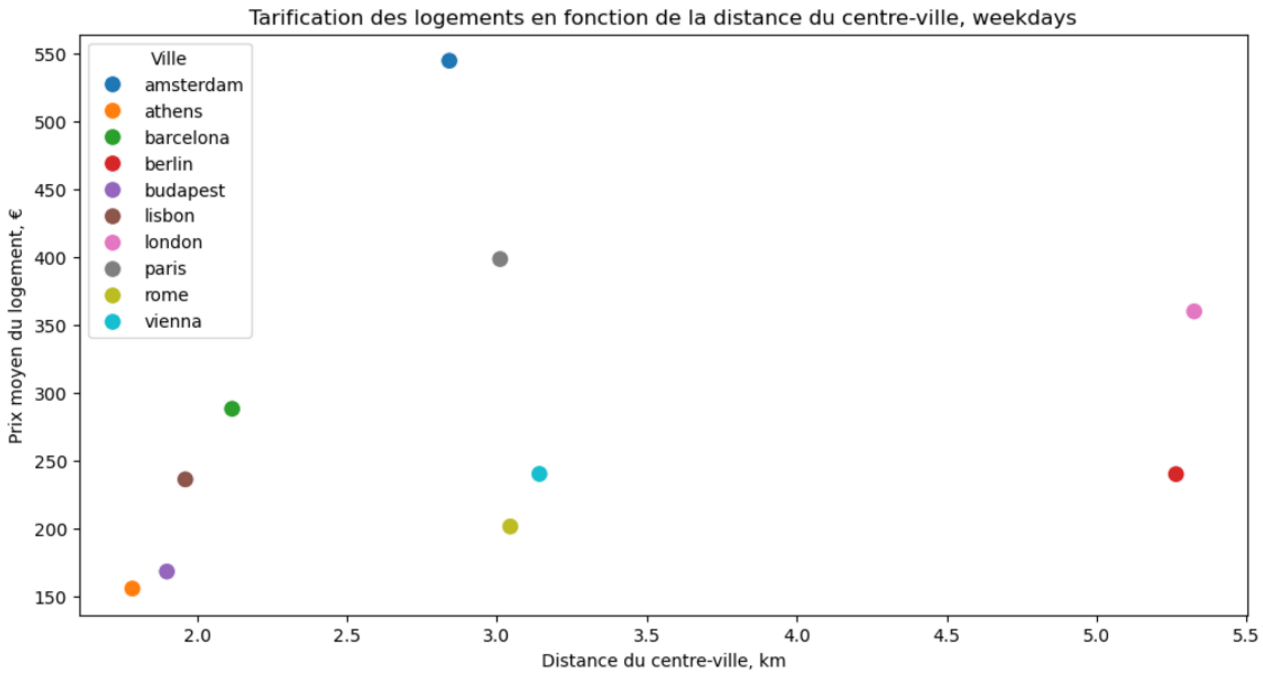


FIGURE 8 – Tarification des logements en fonction de la distance du centre-ville en semaine

Ce graphique montre la relation entre la distance moyenne des logements par rapport au centre-ville et leur prix moyen, pour plusieurs grandes villes européennes, en semaine.

Par exemple, **Amsterdam** affiche des prix très élevés, même pour des logements situés à une distance relativement importante du centre-ville, reflétant une forte demande et un marché immobilier tendu.

En revanche, **Athènes** présente des prix bien plus accessibles, même pour des logements proches du centre, ce qui peut témoigner d'une moindre pression sur le marché.

Pour **Lisbonne**, on observe une diminution des prix moyens à mesure que l'on s'éloigne du centre-ville, typique des villes où le centre concentre l'attractivité principale.

### c) Relation entre la tarification des logements et la distance de la station de métro

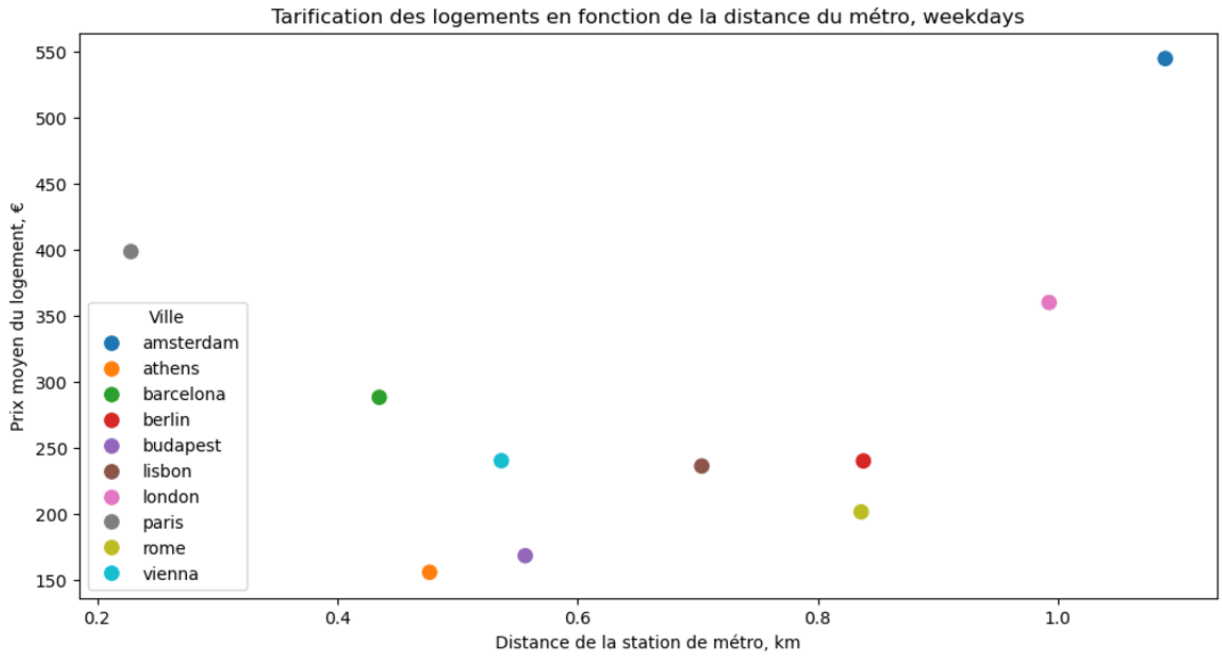


FIGURE 9 – Tarification des logements en fonction de la distance du métro en semaine

Ce graphique représente la distance moyenne des logements par rapport à la station de métro la plus proche en fonction de leur prix moyen, pour plusieurs grandes villes européennes pendant les jours de la semaine.

Par exemple, **Amsterdam** affiche des prix très élevés, même pour des logements situés à proximité des stations de métro, reflétant un marché immobilier très tendu.

En revanche, **Athènes** et **Budapest** présentent des prix nettement plus accessibles, malgré une bonne proximité des logements aux stations de métro, soulignant une plus grande accessibilité financière dans ces villes.

Pour **Lisbonne**, on observe une augmentation des prix même à faible distance des stations de métro, ce qui indique une forte valorisation de l'accès aux transports publics dans cette ville.

**d) Diagramme circulaire (pie chart) de l'indice d'attraction de l'emplacement du logement**

| Ville | Indice |
|---|---|
| Amsterdam | 271.009899 |
| Athens | 155.267315 |
| Barcelona | 464.371805 |
| Berlin | 109.798325 |
| Budapest | 212.778452 |
| Lisbon | 223.001652 |
| London | 296.466825 |
| Paris | 372.177841 |
| Rome | 465.529261 |
| Vienna | 123.079791 |

TABLE 3 – Indice d'attraction de l'emplacement pour chaque ville

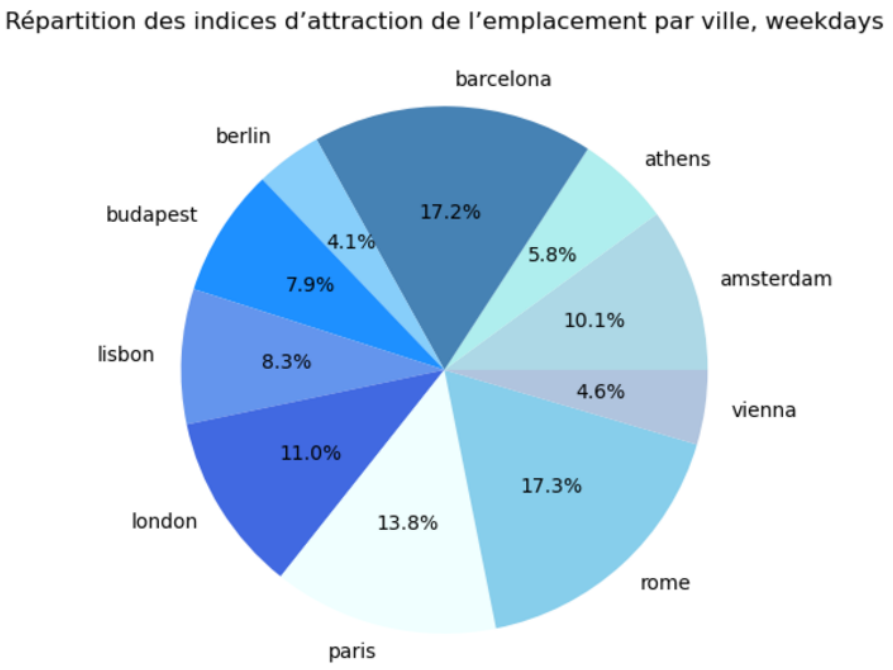


FIGURE 10 – Pie chart de répartition des indices d'attraction de l'emplacement

Chaque secteur dans le diagramme représente une ville. La taille de chaque secteur indique la proportion de l'indice d'attraction de l'emplacement pour cette ville par rapport à l'ensemble des indices d'attraction de toutes les villes. Plus précisément, un secteur plus large correspond à une ville ayant un indice d'attraction plus élevé.

Dans le Tableau 4, Rome et Barcelone sont les villes avec l'indice d'attraction de l'emplacement le plus élevé (en moyenne, 465.53 et 464.37 respectivement). C'est ce que l'on note aussi dans la Figure 9, avec les pourcentages les plus élevés de la proportion de l'indice par rapport au total. Cela signifie que ces villes sont plus attractives en termes d'emplacement.

Cependant, l'indice d'attraction de Berlin représente 4.1% de la somme des indices d'attraction de toutes les villes. Ces pourcentages permettent d'appréhender rapidement la part relative de chaque ville dans l'indice global.

### e) Régression linéaire

Regarder la régression linéaire est essentiel puisqu'elle permet d'analyser la relation entre une variable dépendante (`realSum`) et des variables indépendantes (`dist`, `metro_dist`, `attr_index`). Cela aide à identifier les facteurs qui influencent significativement et à prédire le comportement des prix en fonction des valeurs des variables explicatives. Grâce aux TDs de Machine Learning sur Python, on peut l'appliquer. Voici le modèle économétrique ajusté :

$$\text{realSum} = \beta_0 + \beta_1 \cdot \text{dist} + \beta_2 \cdot \text{metro\_dist}$$

TABLE 4 – Résultats de la régression OLS

| **Variable** | **Coefficient** | **Erreur standard** | **t** | **P>\|t\|** | **[0.025, 0.975]** |
|---|---|---|---|---|---|
| const | 164.4443 | 124.379 | 1.322 | 0.228 | [-129.666, 458.555] |
| Dist | 9.5192 | 37.709 | 0.252 | 0.808 | [-79.648, 98.687] |
| Metro | 134.7111 | 181.023 | 0.744 | 0.481 | [-293.339, 562.761] |

| | | | |
|---|---|---|---|
| **Omnibus :** | 2.578 | **Durbin-Watson :** | 2.132 |
| **Prob(Omnibus) :** | 0.276 | **Jarque-Bera (JB) :** | 1.543 |
| **Skew :** | 0.919 | **Prob(JB) :** | 0.462 |
| **Kurtosis :** | 2.433 | **Cond. No. :** | 17.0 |

Les résultats de la régression montrent que l'intercept (`const`) est de 164.44, ce qui signifie qu'en l'absence des variables explicatives (`dist` et `metro_dist`), le prix prévu d'un logement est de 164,44 euros (€). Cependant, ce coefficient n'est pas significatif (p-value = 0.228) à 10 %, il n'est pas statistiquement différent de zéro.

Pour la variable `dist` (la distance du centre-ville), le coefficient est de 9.52, ce qui indique qu'une augmentation supplémentaire de 1 km de distance est associée à une augmentation moyenne du prix de 9,52 euros (€). Toutefois, ce coefficient est également non significatif (p-value = 0.808).

Pour la variable `metro_dist` (la distance de la station de métro), le coefficient est de 134.71, indiquant que 1 km supplémentaire provoque une augmentation moyenne du prix de 134,71 euros (€). Cependant, ce coefficient n'est pas significatif non plus (p-value = 0.481).

Le coefficient de détermination (R-squared) montre que seulement 13,3% de la variance des prix des logements est expliquée par les variables `dist` et `metro_dist`. Cela indique que le modèle économétrique pourrait être trop simple ou mal adapté aux données, surtout avec un faible nombre d'observations (N=10), comme on prend les prix moyens.

On peut conclure que ni la distance au centre-ville ni la proximité au métro n'ont pas une influence significative sur la tarification des logements dans ce modèle.

## I.5 Question 4 : L'impact des dynamiques sociales, notamment l'effet des évaluations, popularité et caractéristiques des hôtes (superhôte) sur les prix

D'abord, il faut définir les variables qu'on peut associer à l'effet des évaluations, popularité et caractéristiques des hôtes (superhôte) sur les prix.

- `host_is_superhost` : Variable binaire indiquant si l'hôte est un « Superhôte » sur Airbnb (`boolean`)
- `multi` : Variable binaire indiquant si l'annonce appartient à des hôtes ayant 2 à 4 offres (`integer`)
- `biz` : Variable binaire indiquant si l'annonce appartient à des hôtes ayant plus de 4 offres (`integer`)
- `guest_satisfaction_overall` : Note globale de l'annonce par les clients (`float`)
- `attr_index_norm` : Indice d'attraction normalisé (échelle 0-100) (`float`)
- `cleanliness_rating` : Note de propreté (`float`)

Ainsi, la variable de prix total de l'hébergement *realSum* correspond à la tarification et est de type `integer`. Toutes les autres variables associées sont de type `float`.

Il y a plusieurs manières d'explorer l'impact des dynamiques sociales sur les prix. Comme dans la section I.4, on fait quelques analyses principaux, comme l'analyse des corrélations et la régression linéaire, pour mieux comprendre les tendances et les relations entre les variables "sociales" et le prix. Voici quelques idées :

### a) Analyse des corrélations

| **Variables** | **realSum** | **host_is_superhost** |
|---|---|---|
| realSum | 1.000000 | -0.504611 |
| host_is_superhost | -0.504611 | 1.000000 |

TABLE 5 – Corrélation entre `realSum` et `host_is_superhost`

| **Variables** | **realSum** | **multi** |
|---|---|---|
| realSum | 1.000000 | -0.203259 |
| multi | -0.203259 | 1.000000 |

TABLE 6 – Corrélation entre `realSum` et `multi`

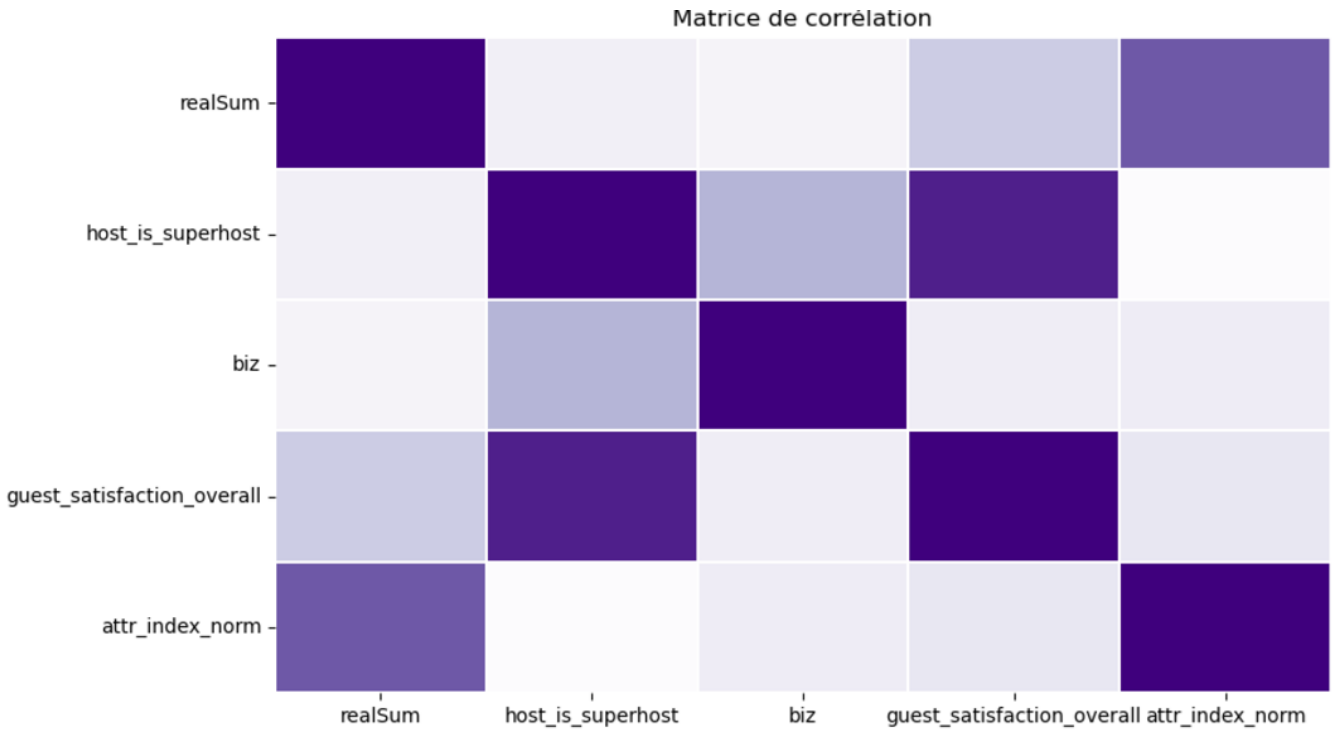


FIGURE 11 – Matrice de corrélation

| Variables | realSum | biz |
|---|---|---|
| realSum | 1.000000 | -0.570504 |
| biz | -0.570504 | 1.000000 |

TABLE 7 – Corrélation entre `realSum` et `biz`

| Variables | realSum | guest_satisfaction_overall |
|---|---|---|
| realSum | 1.000000 | -0.165855 |
| guest_satisfaction_overall | -0.165855 | 1.000000 |

TABLE 8 – Corrélation entre `realSum` et `guest_satisfaction_overall`

| Variables | realSum | attr_index_norm |
|---|---|---|
| realSum | 1.000000 | 0.539408 |
| attr_index_norm | 0.539408 | 1.000000 |

TABLE 9 – Corrélation entre `realSum` et `attr_index_norm`

Les résultats de la matrice de corrélation montrent plusieurs relations intéressantes entre les variables.

D'abord, il existe une corrélation négative modérée entre le statut de superhôte (`host_is_superhost`) et le prix des logements (-50,5%). Cela suggère que les superhôtes ont tendance à proposer des prix légèrement plus bas que les hôtes non-superhôtes. Cela pourrait être dû à un plus grand volume de réservations ou à une stratégie de prix plus compétitive de la part des superhôtes pour maintenir leur statut.

Ensuite, la corrélation entre la variable `multi` (qui indique si un hôte a plusieurs annonces) et le prix est faible mais négative aussi (-20,3%). Cela indique que les hôtes avec plusieurs annonces peuvent avoir une tendance à proposer des prix plus bas, probablement pour attirer davantage de clients ou pour maintenir une politique de prix compétitive.

Ainsi, la corrélation négative plus forte existe entre la variable `biz` (qui indique si un hôte a plus de 4 annonces) et le prix des logements (-57%). Cela pourrait signifier que les hôtes ayant un grand nombre d'annonces adoptent des stratégies de tarification plus agressives, en offrant des prix plus bas pour maximiser leur taux de remplissage.

Faible corrélation négative entre `guest_satisfaction_overall` et `realSum` (-16,6%) signifie que des évaluations de satisfaction légèrement meilleures ne sont pas nécessairement associées à des prix plus élevés. Cependant, cette relation faible indique que la satisfaction des invités n'est pas un facteur majeur dans la tarification des logements dans cette analyse.

Cependant, l'indice d'attractivité normalisé (`attr_index_norm`) et `realSum` ont la corrélation positive modérée (54%). Cela indique que les logements situés dans des zones plus attractives (selon l'indice d'attractivité) ont tendance à être plus chers, ce qui est logique étant donné que les zones plus populaires et demandées attirent des prix plus élevés.

### b) Régression linéaire multiple

Le modèle économétrique ajusté est :

$$\text{realSum} = \beta_0 + \beta_1 \cdot \text{host\_is\_superhost} + \beta_2 \cdot \text{multi} + \beta_3 \cdot \text{biz} + \beta_4 \cdot \text{guest\_satisfaction\_overall}$$

Table 10 – Résultats de la régression OLS

| Variable | Coefficient | Erreur standard | t | P>\|t\| | [0.025, 0.975] |
|---|---|---|---|---|---|
| const | 14500.0000 | 9661.966 | 1.501 | 0.194 | [-10300.000, 39300.000] |
| `host_is_superhost` | 1613.1391 | 1610.957 | 1.001 | 0.363 | [-2527.958, 5754.236] |
| `multi` | -1423.0599 | 920.058 | -1.547 | 0.183 | [-3788.144, 942.024] |
| `biz` | -1494.5257 | 691.706 | -2.161 | 0.083 | [-3272.612, 283.561] |
| `guest_sat_overall` | -147.7377 | 103.782 | -1.424 | 0.214 | [-414.518, 119.043] |

| | | | |
|---|---|---|---|
| **R-squared :** | 0.723 | **Adj. R-squared :** | 0.501 |
| **F-statistic :** | 3.255 | **Prob(F-statistic) :** | 0.114 |
| **Log-Likelihood :** | -55.588 | **AIC :** | 121.2 |
| **BIC :** | 122.7 | **No. Observations :** | 10 |
| **Df Residuals :** | 5 | **Df Model :** | 4 |
| **Durbin-Watson :** | 1.840 | **Omnibus :** | 3.078 |
| **Prob(Omnibus) :** | 0.215 | **Jarque-Bera (JB) :** | 1.305 |
| **Skew :** | 0.885 | **Prob(JB) :** | 0.521 |
| **Kurtosis :** | 2.974 | **Cond. No. :** | 3.26e+04 |

Les résultats de la régression montrent que le modèle explique 72,3 % de la variance des prix des logements (`R-squared` = 0.723), ce qui indique une bonne capacité explicative du modèle. Cependant, `Adj. R-squared` de 0.553 suggère que ce modèle est moins performant quand on tient compte de la taille de l'échantillon, qui est relativement petite (`N` = 10 observations).

En ce qui concerne les coefficients, le statut de superhôte (`host_is_superhost`) augmente le prix de 1613,14 euros (€), mais cet effet n'est pas statistiquement significatif (p-value de

0.291). De même, les hôtes avec plusieurs annonces (`multi`) ont tendance à fixer des prix plus bas (-1423,1 euros (€)), mais cet effet n'est également pas significatif (p-value = 0.137).

Pour les hôtes ayant plus de 4 annonces (`biz`), l'effet est négatif (-1494,53 euros (€)) et est marginalement significatif à 10% (p-value de 0.061). Quant à la satisfaction des clients (`guest_sat_overall`), une augmentation de la note globale semble légèrement réduire le prix de -147,74 euros, mais cet effet n'est pas significatif non plus (p-value = 0.134).

Ces résultats suggèrent qu'il pourrait être nécessaire d'augmenter la taille de l'échantillon pour tirer des conclusions plus fiables sur l'impact des variables sociales sur les prix des logements.

### c) Pie chart de l'indice d'attraction normalisé du logement par ville

Comme pour le pie chart de la section I.4, chaque secteur dans le diagramme représente une ville et la taille de chaque secteur indique la proportion de l'indice d'attraction normalisé pour cette ville par rapport à l'ensemble des indices de toutes les villes. Un pourcentage plus élevé correspond à une ville ayant un indice d'attraction plus large.

TABLE 11 – Valeurs normalisées de l'indice d'attractivité par ville

| **Ville** | **Indice d'attractivité normalisé** |
|---|---|
| Amsterdam | 14.246499 |
| Athens | 5.740839 |
| Barcelona | 16.636220 |
| Berlin | 16.803111 |
| Budapest | 12.675248 |
| Lisbon | 7.324730 |
| London | 20.537398 |
| Paris | 18.204358 |
| Rome | 10.426968 |
| Vienna | 8.762474 |

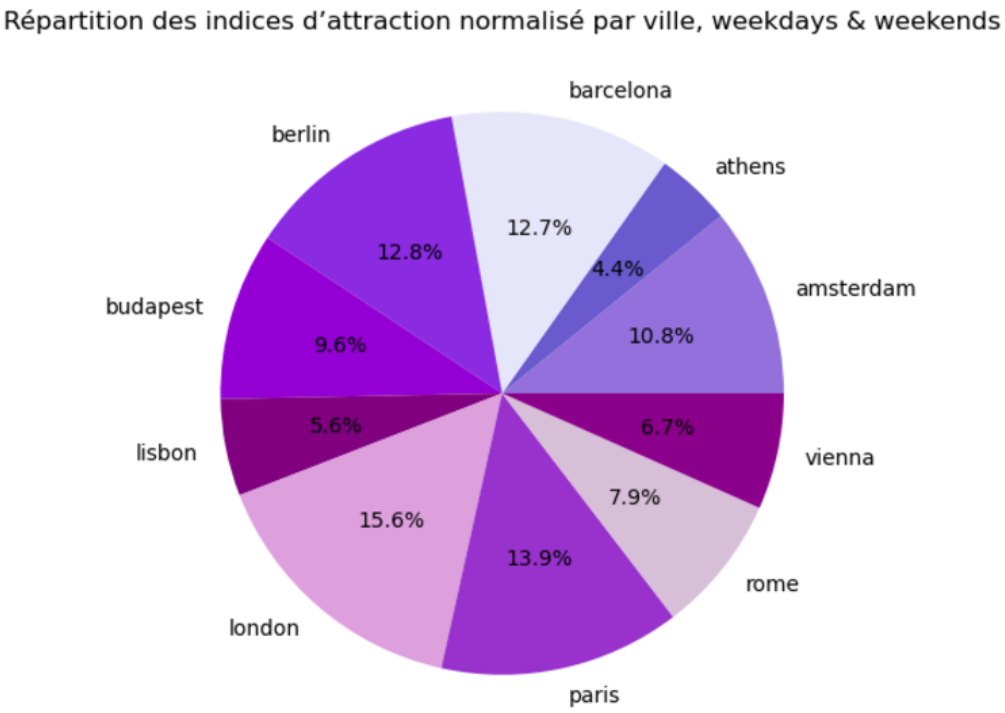


FIGURE 12 – Pie chart de répartition des indices d'attraction normalisés

La Figure 10 montre que l'indice d'attraction normalisé de Londres représente 15.6% de la somme des indices d'attraction normalisé de toutes les villes. Cela signifie que cette ville est plus attractive.

### d) Relation entre satisfaction des clients et prix du logement par ville

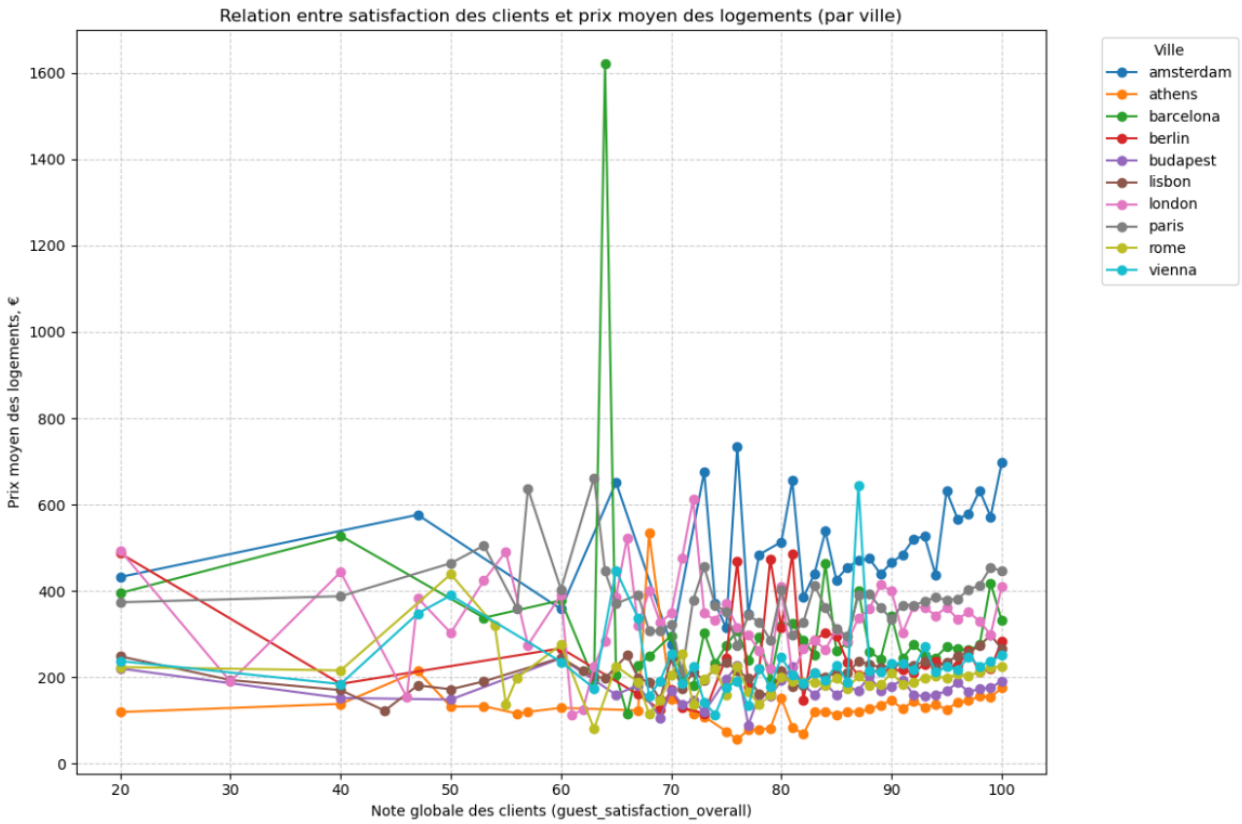


FIGURE 13 – Relation entre les prix et notes globales

Les données montrent une dispersion importante, suggérant qu'il n'existe pas de corrélation systématique entre le prix et la satisfaction des clients. **Amsterdam** et **Londres** affichent des prix moyens nettement plus élevés, indépendamment de la note de satisfaction, ce qui reflète des marchés immobiliers compétitifs. **Paris**, **Vienne**, **Lisbonne**, et **Barcelone** présentent des prix modérés et stables, avec une légère augmentation pour les meilleures notes de satisfaction, traduisant des marchés équilibrés. À l'inverse, des villes comme **Athènes**, **Budapest**, et **Rome** maintiennent des prix plus accessibles, même pour des niveaux de satisfaction élevés, traduisant un coût de vie inférieur.

Des anomalies, comme un pic extrême pour une note de satisfaction autour de 60, pourraient refléter des logements haut de gamme ou des spécificités locales. Certaines fluctuations, notamment visibles pour **Lisbonne**, **Rome**, et **Vienne**, montrent des variations irrégulières dans les prix, ce qui pourrait être lié à des offres particulières ou des marchés moins homogènes.

Globalement, un prix élevé ne garantit pas une satisfaction proportionnelle, et vice versa. La satisfaction semble davantage influencée par des facteurs comme la localisation, les équipements ou l'expérience globale, reflétant les particularités des marchés locaux.

### e) Relation entre la note de propreté et le prix du logement

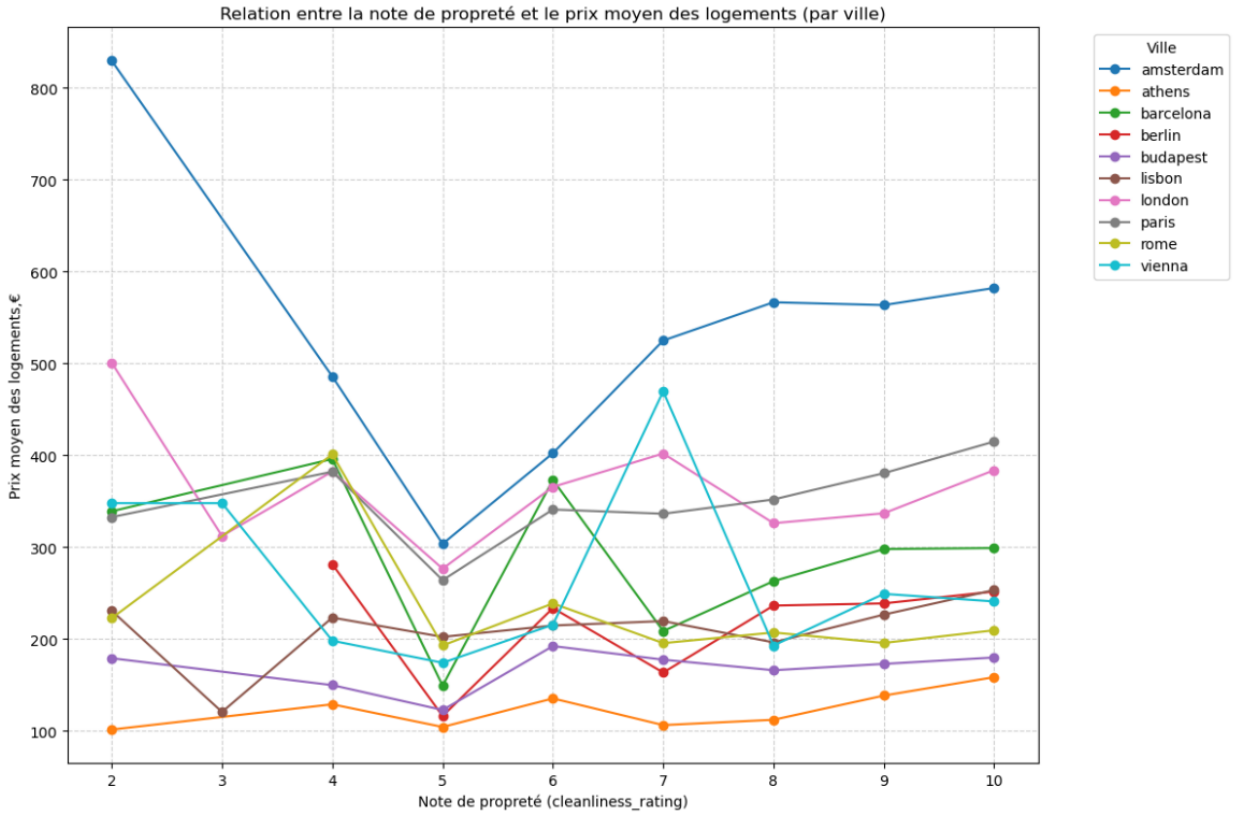


FIGURE 14 – Relation entre la note de propreté et le prix moyen des logements

Une tendance générale se dégage : les prix ont tendance à augmenter avec des notes de propreté plus élevées, bien que des variations importantes soient visibles selon les villes.

**Amsterdam** et **Londres** se distinguent par des prix nettement plus élevés, quel que soit le niveau de propreté, ce qui reflète des marchés immobiliers très compétitifs. **Paris** et **Vienne**, quant à elles, affichent une évolution plus modérée, avec des prix moyens stables ou légèrement croissants pour les meilleures notes de propreté. À l'inverse, des villes comme **Athènes** et **Budapest** maintiennent des prix bien plus accessibles, même pour des notes de propreté élevées, traduisant un coût de vie inférieur.

Des anomalies sont également visibles : par exemple, certaines villes, comme **Lisbonne** ou **Rome**, montrent des variations irrégulières, avec des fluctuations de prix pour des notes de propreté moyennes, notamment autour de 5 et 6. Cela peut s'expliquer par des caractéristiques spécifiques des logements ou des facteurs locaux influençant les prix.

Globalement, bien que la propreté joue un rôle dans la tarification, d'autres éléments comme la localisation, les équipements ou la demande locale restent déterminants dans la fixation des prix.

# II. Méthodologie
# Les étapes de développement et les bibliothèques utilisées

### Description des étapes de développement

Pour mener à bien ce projet, nous avons suivi méthodiquement les étapes décrites dans la consigne. Dans un premier temps, nous avons essayé de bien comprendre le but final du projet afin de déterminer les objectifs à atteindre. Une fois cela clarifié, nous avons réfléchi à la meilleure manière de procéder et à structurer nos tâches pour progresser efficacement.

## Outils, bibliothèques utilisées, etc.

Nous avons utilisé **GitHub** pour centraliser nos fichiers et codes, ce qui nous a permis de travailler en parallèle tout en synchronisant nos contributions. Cette approche a facilité la collaboration et l'intégration de chaque partie du projet. La Figure 15 montre notre espace de travail, dans lequel on a géré les notebooks et les bases de données.


FIGURE 15 – Espace de travail GitHub

Côté technique, plusieurs bibliothèques Python ont été utilisées :

- **pandas** : manipulation des Series et DataFrames et analyse des données ;
- **numpy** : manipulation des tableaux multidimensionnels et calculs numériques ;
- **os** : travail avec les fichiers et les chemins ;
- **seaborn** et **matplotlib** : visualisation des données ;
- **statsmodels** : analyses statistiques et régression ;
- **tkinter** : création d'interfaces graphiques simples.

## Répartition du travail dans le groupe

Bien que nous ayons chacun eu une partie attribuée, nous avons beaucoup travaillé ensemble et nous nous sommes entraidés tout au long du projet. Cela nous a permis d'assurer une cohérence globale et de résoudre rapidement les éventuels problèmes rencontrés.

Nous avons également organisé plusieurs réunions via Zoom pour discuter de nos avancées, partager nos idées et ajuster notre plan de travail si nécessaire.

Voici la répartition initiale des tâches :

— **Partie 1** : Anastasiia & Victoria – cette partie s'est concentrée sur la collecte, l'analyse des données et leur visualisation.

— **Partie 2** : Théo – création du moteur de recherche.

— **Partie 3** : Moheb – responsable de l'intégration des résultats et de la finalisation de l'interface utilisateur.

— **Rédaction LaTex** : Anastasiia & Victoria

# III. Conception
# Architecture du programme et de l'interface

## Architecture de la partie logiciel du projet

### Fichiers principaux

Le projet est organisé dans un dépôt **GitHub** structuré pour garantir une collaboration efficace et une navigation claire. Ce dépôt contient principalement deux dossiers et les notebooks sur lesquels on a codé :

— Dossier **data** : Ce dossier regroupe l'ensemble des datasets nécessaires à l'exécution du programme. Les données brutes et prétraitées y sont stockées pour un accès centralisé.

— Dossier **src** : Ce dossier contient un module regroupant quelques fonctions clés utilisées dans le programme

— Des notebooks dédiés à la Partie 1 (collecte et préparation des données) et aux Parties 2-3 (analyses et visualisations finales).

### Interactions des scripts

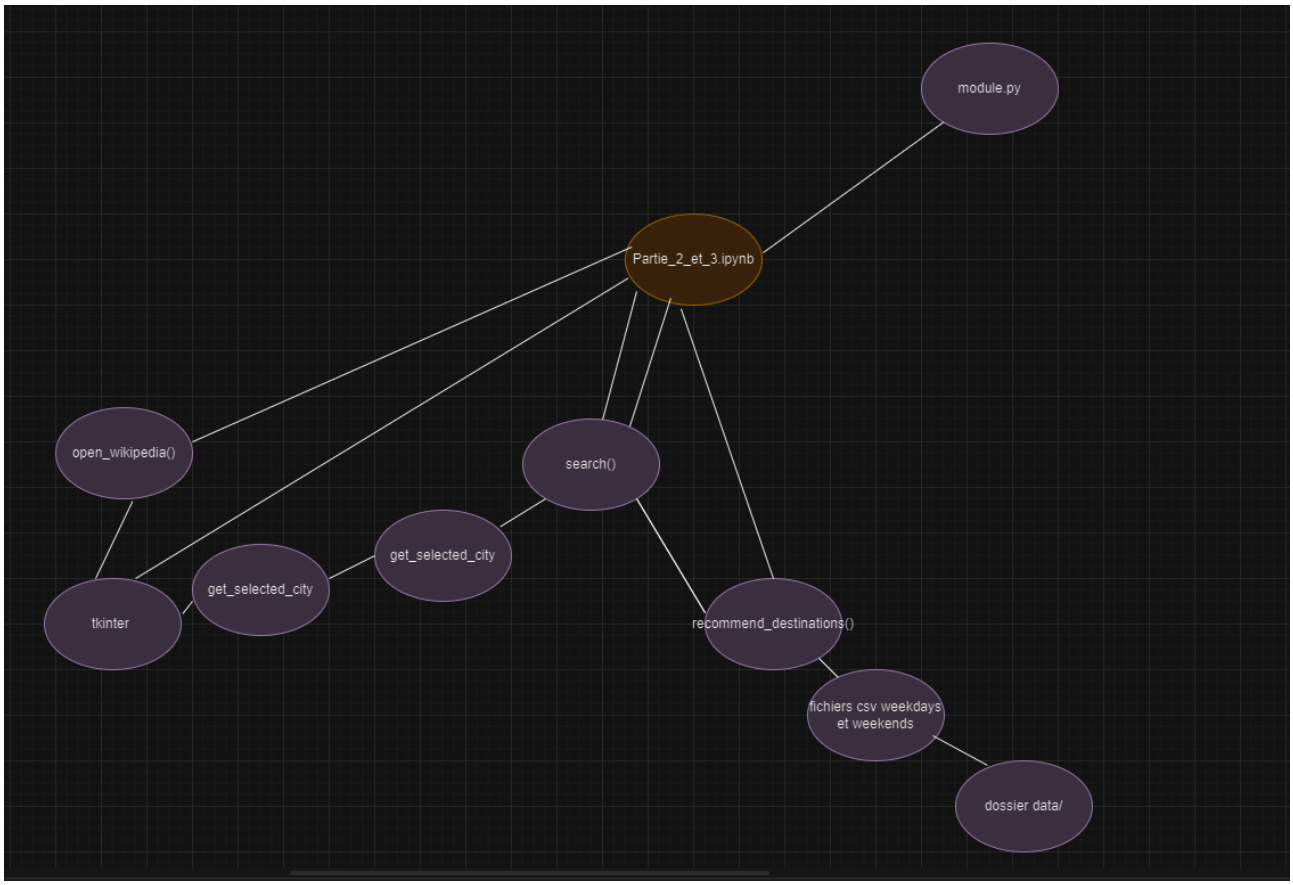


FIGURE 16 – Schema des interactions

Le notebook **Partie_2_et_3.ipynb** interagit avec :

- *L'interface graphique tkinter*, qui gère l'interaction avec l'utilisateur pour collecter les critères de recherche et afficher les résultats ;
- Les fichiers CSV qu'il charge et traite pour générer des recommandations grâce à *recommend_destinations()* ;
- Les fonctions :
  - *recommend_destination()* : applique des filtres en fonction des critères saisis (budget, capacité, distance, etc.) ;
  - *search()* : récupère les critères de l'utilisateur à partir de *l'interface tkinter* et utilise *recommend_destination* pour traiter les données, puis affiche les résultats dans un tableau ;
  - *open_wikipedia()* : ouvre une page Wikipedia concernant la ville sélectionnée dans un navigateur ;
  - *get_selected_city()* : identifie la ville choisie par l'utilisateur dans le tableau des résultats pour ouvrir la page Wikipedia associée.

**Déroulement des actions du programme**

Le programme suit un enchaînement logique d'actions pour répondre aux besoins de l'utilisateur :

1. **Chargement des données** : Les datasets nécessaires sont importés et préparés pour être exploités dans les étapes suivantes.
2. **Saisie des caractères par l'utilisateur** : Grâce à une interface graphique intuitive, l'utilisateur peut fournir les paramètres nécessaires, tels que des critères de filtrage ou des préférences.
3. **Recommandations et résultats** : Les données sont filtrées en fonction des critères saisis, et les résultats sont affichés sous forme de tableau trié par pertinence, offrant ainsi une visualisation claire et exploitable.

## Interface utilisateur du programme

L'interface utilisateur a été conçue pour être simple et fonctionnelle. Elle permet une interaction fluide entre l'utilisateur et le programme, grâce à des champs de saisie et des boutons clairement identifiés. L'interface utilisateur du programme se divise en deux sections principales : **les critères de recherche** et **les résultats des recommandations**.

— **Section critères** ***(1)***

Dans **la section des critères de recherche**, l'utilisateur peut saisir ses préférences via des champs dédiés. Les critères incluent **le budget maximal** (€), **la capacité minimale** (en nombre de personnes), **la distance maximale** (en km) par rapport au centre-ville, ainsi que **le type de séjour** (jours de semaine ou week-ends), sélectionnable via un menu déroulant.

Une fois ces informations renseignées, l'utilisateur peut cliquer sur le bouton **Rechercher**, qui déclenche la fonction associée *search*. Cette fonction récupère les données saisies, applique les filtres correspondants grâce à la fonction *recommend_ destinations* et génère une liste des villes recommandées en fonction des critères donnés (cf. Figure 17).

— **Section résultats** ***(2)***

**La section des résultats** affiche les recommandations sous forme de tableau interactif. Ce tableau présente les statistiques des villes répondant aux critères, incluant **le prix moyen** (€), **le prix minimum** (€), **le prix maximum** (€), ainsi que des indicateurs de qualité comme **la propreté** (/ 10) et **la satisfaction client** (/ 100). Le bouton "voir sur Wikipedia est aussi afficher dans cette fenêtre pour rediriger l'utilisateur sur la page wikipedia de la ville selectionnée. Les données sont triées par prix moyen pour mettre en avant les options les plus économiques. Si aucune ville ne correspond aux critères, un message d'information est affiché pour inviter l'utilisateur à ajuster sa recherche (cf. Figure 17).

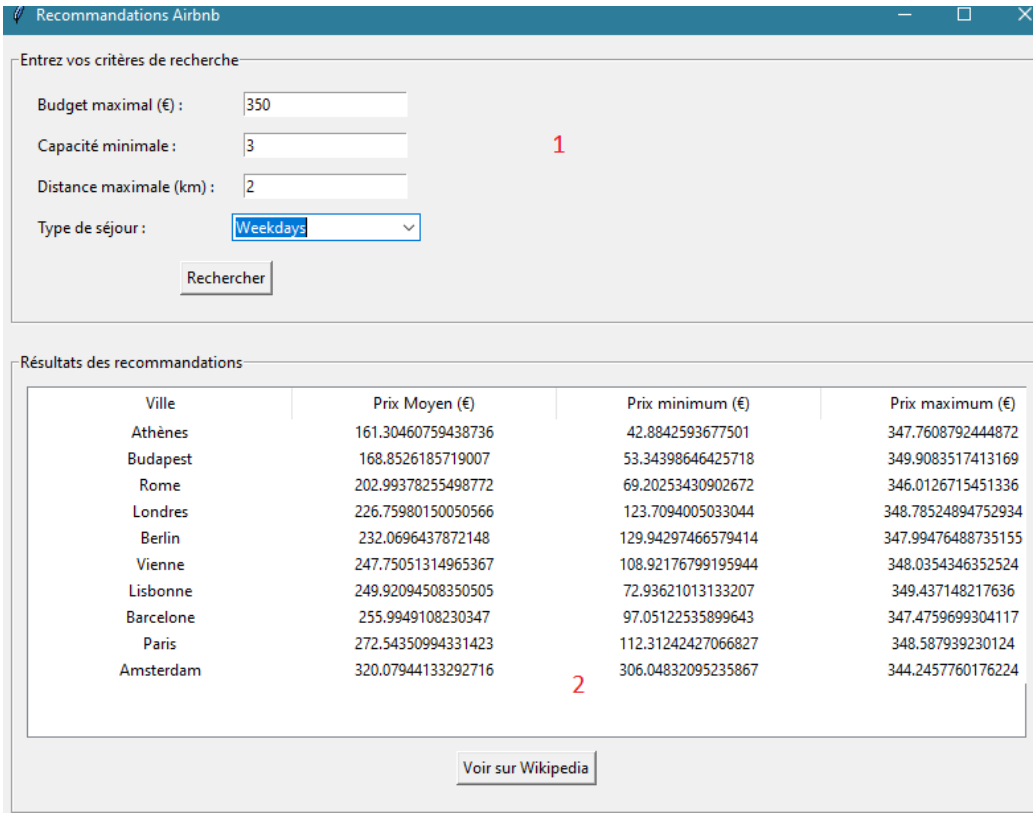


FIGURE 17 – Rendu graphique finale

# IV. Implémentation
# Fonctionnalités développées et exemples de code pertinents

Trois fonctions importantes ont été dévéloppées :

— **Fonction de recommandation**

Cette fonction filtre les données en fonction des critères saisis par l'utilisateur et renvoie les résultats sous forme de DataFrame.

```
def recommend_destinations(budget=None, capacite=None, distance=None,
    travel_type=None):
    filtered_data = all_data.copy()

    if budget is not None:
        filtered_data = filtered_data[filtered_data["realSum"] <=
            budget] # filtres
    if capacite is not None:
        filtered_data = filtered_data[filtered_data["person_capacity"]
            >= capacite]
    if distance is not None:
        filtered_data = filtered_data[filtered_data["dist"] <= distance
            ]
    if travel_type:
        filtered_data = filtered_data[filtered_data["Type"].str.lower()
            == travel_type.lower()]

    # Groupement par ville et calcul des statistiques :
    ville_stats = filtered_data.groupby("City").agg({
```

```
        "realSum": ['mean', 'min', 'max'], #moyenne, min et max des prix
        "cleanliness_rating": 'mean',  #moyenne de la note de propreté
        "guest_satisfaction_overall": 'mean'  #moyenne de la satisfaction overall
    }).reset_index()

    ville_stats.columns = ["Ville", "Prix Moyen (euro)", "Prix minimum (euro)", "Prix maximum (euro)", #on renomme les colonnes
                           "Propreté, sur 10", "Satisfaction client, sur 100"]

    ville_stats = ville_stats.sort_values(by="Prix Moyen (euro)").head(10)  # on tri par le prix
    return ville_stats
```

Listing 1 – Fonction `recommend_destinations()`

— **Fonction de recherche**

Elle effectue une recherche basée sur des critères donnés par l'utilisateur, on y utilise la fonction de recommandation montrée au dessus et affiche les résultats dans une interface graphique sous forme de tableau.

```
def search():
    try:
        # Récupérer les valeurs des entrées utilisateur
        budget = float(budget_entree.get()) if budget_entree.get() else None  # Entrée utilisateur pour budget
        capacite = int(capacite_entree.get()) if capacite_entree.get() else None
        distance = float(distance_entry.get()) if distance_entry.get() else None
        travel_type = travel_type_var.get()  # Type de voyage choisi

        # Générer les recommandations
        recommendations = recommend_destinations(budget, capacite, distance, travel_type)

        # Afficher les résultats dans le tableau
        for row in tree.get_children():  # Supprimer les lignes actuelles
            tree.delete(row)
        for _, row in recommendations.iterrows():  # Ajouter chaque ligne des recommandations
            tree.insert("", "end", values=list(row))

        # Gérer les cas ou aucune recommandation n'est trouvée
        if recommendations.empty:
            messagebox.showinfo("Recommandations", "Aucune destination ne correspond à vos critères.")
    except Exception as e:
        # Afficher une erreur en cas de problème
        messagebox.showerror("Erreur", f"Une erreur s'est produite : {e}")
```

Listing 2 – Fonction `search()`

— **Fonction Wikipedia**

On a introduit une fonction supplémentaire wikipedia qui permet de diriger l'utilisateur vers une page Wikipedia qui décrit une ville sélectionnée dans un navigateur.

```
def open_wikipedia(city):
    wikipedia_links = {
        "London": "https://fr.wikipedia.org/wiki/London",
        "Paris": "https://fr.wikipedia.org/wiki/Paris",
        "Rome": "https://fr.wikipedia.org/wiki/Rome",
        "Vienne": "https://fr.wikipedia.org/wiki/Vienne_(Autriche)",
        "Amsterdam": "https://fr.wikipedia.org/wiki/Amsterdam",
        "Athènes": "https://fr.wikipedia.org/wiki/Athènes",
        "Barcelone": "https://fr.wikipedia.org/wiki/Barcelone",
        "Berlin": "https://fr.wikipedia.org/wiki/Berlin",
        "Budapest": "https://fr.wikipedia.org/wiki/Berlin",
        "Lisbonne": "https://fr.wikipedia.org/wiki/Lisbonne",
    }
    if city in wikipedia_links:
        webbrowser.open(wikipedia_links[city])
    else:
        messagebox.showinfo("Info", f"Aucun lien disponible pour {city}
            ")
```

Listing 3 – Fonction `open_wikipedia()`

# V. Tests et Validation
# Méthodes de test appliquées au projet

Pour vérifier le bon fonctionnement du programme, nous exécutions des morceaux de code afin d'identifier d'éventuelles erreurs. En cas de problème, nous analysions son origine pour le résoudre. Nous utilisions WhatsApp pour collaborer, nous entraider sur les bugs et valider le travail des uns et des autres.

Pour la partie liée à Tkinter, nous avons suivi des tutoriels afin de comprendre le fonctionnement de la bibliothèque et de nous y familiariser. Nous avons dû réécrire la fonction de recherche pour la rendre compatible avec Tkinter ; ainsi, au lieu d'utiliser des *"input"*, nous avons opté pour *".get()"*. Pour tester les fonctions de recherche et de recommandation, nous les avons d'abord exécutées avec des données de test afin de vérifier si l'output correspondait bien au résultat attendu.

# Perspectives et Conclusion

### Si le projet était à refaire

Nous avions plusieurs idées sur la manière de développer le projet ou de retravailler certaines parties.

D'abord, pour l'étude des effets influant sur les prix, on aurait pu faire une nettoyage des données, s'il existe des variables NaN. Ainsi, du point de vue économétrique, dans l'analyse des corrélations, on a obtenu les relations ambiguée, notamment les corrélations positives entre les distances et les prix des logements et le coefficient de détermination peu élevé. Notre analyse n'a pas démontré les résultats attendus. La relation entre les prix et les variables dites "impacts" est manifestement complexe, influencée par une multitude de facteurs géographiques, saisonniers, sociaux et même politiques. Néanmoins, des pistes d'amélioration pour les modèles analysés se dessinent. Par exemple, intégrer des données de panel et augmenter le nombre d'observations pourrait permettre de mieux capturer les variations significatives des variables dans le temps. En revanche, travailler uniquement avec des données en coupe transversale risque de passer à côté des effets sociaux-géographiques ou des impacts liés à des chocs économiques ou politiques dans un pays.

Pour la création du moteur de recherche et de son interface graphique, nous aurions pu utiliser la bibliothèque customtkinter, très proche de tkinter mais offrant une apparence plus moderne. De plus, concevoir dès le départ une fonction de recherche directement adaptée à tkinter aurait permis d'optimiser le processus de développement.

Enfin, nous aurions pu intégrer des fonctionnalités supplémentaires, comme la recherche de restaurants, de transferts aéroport, ou de lieux touristiques populaires bien notés à proximité du logement, en supposant que la position géographique de l'Airbnb soit disponible.

**Qu'avons nous appris et diffcultés rencontrées**

**Théo :**

- **Ce que j'ai appris grâce à ce projet :**
  - Manipuler de grosses quantités de données grâce à la bibliothèque pandas
  - Fusionner plusieurs bases de données pour en former qu'une de façon cohérente
  - Vérifier l'absence de valeurs manquantes dans les jeux de données
  - Structurer le code avec des fonctions réutilisables
  - Comment classer les résultats (par prix croissant et satisfaction décroissante)
  - Travailler en groupe (répartition des tâches et utilisation de GitHub)
- **Les difficultés que j'ai rencontrées :**
  - Permettre à l'utilisateur de ne renseigner que certains critères (et ignorer les autres) a compliqué la logique de filtrage.
  - Rendre clair et compréhensible le code pour tout le monde (ajout de commentaires et utilisation de Jupyter Notebook pour détailler le code étape par étape).

**Moheb :**

- **Ce que j'ai appris grâce à ce projet :**
  - Apprendre les utilisations basiques de git dans le but de mettre en place une gestion collaborative d'un code dans le cadre d'un projet à plusieurs
  - Travailler en groupe à distance de manière efficace
  - Création d'une interface graphique
  - Appeler une fonction sur le clique d'un bouton
- **Les difficultés que j'ai rencontrées :**
  - Construire la base d'un projet (sur quel support travailler)
  - Apprendre à utiliser de nouvelles librairies (tkinter)

- Devoir ajuster le code pour le rendre compatible avec la nouvelle librairie (fonction de recherche qu'il fallait modifier pour pouvoir l'utiliser)

**Anastasiia :**

- **Ce que j'ai appris grâce à ce projet :**
  - Créer des DataFrames à partir de dictionnaires (et inversement) et les manipuler : fusionner et filtrer
  - J'ai appris les caractéristiques des structures de données, si elles sont ordonnées, inaltérables ou modifiables
  - Créer des graphiques correspondant à une question posée (histogram, scatter plot, pie chart, corr plot, box plot)
  - Créer une interface graphique
  - Travailler dans Jupyter Notebook (avant j'ai utilisé uniquement Spyder ou Kaggle)
  - Créer des projets et gérer des notebooks sur GitHub
- **Les difficultés que j'ai rencontrées :**
  - Les valeurs empiriques de certaines statistiques, comme les coefficients de corrélation, de détermination et de régression OLS, étaient incohérentes par rapport à ce que on a attendu (Partie 1)
  - L'interprétation correcte de certains graphiques
  - La fusion des données, en prenant en compte les weekdays et les weekends
  - Apprendre à utiliser les blocs try et except pour certaines fonctions
  - Créer des algorithmes théoriques et les exprimer étape par étape dans le code, en corrigeant presque à chaque étape les erreurs

**Victoria :**

- **Ce que j'ai appris grâce à ce projet :**
  - Apprendre à manipuler de grandes quantités de données avec des bibliothèques, créer et gérer des DataFrames à partir de différentes structures (comme des dictionnaires), et fusionner plusieurs bases de données pour les rendre cohérentes.
  - Générer des visualisations graphiques adaptées aux analyses afin de répondre à des questions spécifiques
  - Travailler en collaboration via GitHub pour organiser et suivre un projet, tout en documentant efficacement le code grâce à Jupyter Notebook
  - Acquérir des bases en tests statistiques et les appliquer avec des outils Python
- **Les difficultés que j'ai rencontrées :**
  - Gérer des problèmes liés aux incohérences statistiques rencontrées lors des calculs basés sur des modèles théoriques
  - Adapter le code étape par étape afin de résoudre les incompatibilités avec certaines bibliothèques et assurer son bon fonctionnement
  - Analyser et interpréter les graphiques produits, en vérifiant leur alignement avec les hypothèses et les résultats attendus

Finalement, ce projet nous a aidé à améliorer et approfondir nos compétences en Python, ainsi qu'à découvrir de nouvelles pratiques, à trouver parfois des solutions non conventionnelles et à travailler en groupe en écoutant les idées des autres.

# Bibliographie

## Introduction

[1]. AirBnb. [En ligne]. Consulté à l'adresse : https://www.airbnb.fr/

## Partie 1

[2]. GeeksForGeeks. *Tracer une matrice de corrélation à l'aide de Python* [En ligne]. Consulté à l'adresse : https://www.geeksforgeeks.org/plotting-correlation-matrix-using-python/

[3]. AnalyticsVidhya. *Pie chart in Matplotlib*. [En ligne]. Consulté à l'adresse : https://www.analyticsvidhya.com/blog/2024/02/pie-chart-matplotlib/#:~:text=To%20add%20labels%20and%20percentages,the%20percentages%20should%20be%20displayed

[4]. MatPlotLib *List of named colors*. [En ligne]. Consulté à l'adresse : https://matplotlib.org/stable/gallery/color/named_colors.html

[5]. Pandas *Merge, join, concatenate and compare*. [En ligne]. Consulté à l'adresse : https://pandas.pydata.org/docs/user_guide/merging.html

## Partie 2 - Moteur de recherche

[6]. Python.org. *Erreurs et exceptions*. [En ligne]. Consulté à l'adresse : https://docs.python.org/fr/3/tutorial/errors.html#

[7]. MonCoachData. *Pandas pour le Traitement de Données*. [En ligne]. Consulté à l'adresse : https://moncoachdata.com/blog/pandas-pour-le-traitement-de-donnees/

## Partie 3 - Tkinter

[8]. WayToLearnX. *Label Tkinter*. [En ligne]. Consulté à l'adresse : https://waytolearnx.com/2020/06/label-tkinter-python-3.html

[9]. G. Swinnen. *Apprendre à programmer avec Python 3*. [En ligne]. Consulté à l'adresse : https://inforef.be/swi/download/apprendre_python3_5.pdf

[10]. PythonTutorial. *Tkinter Tutorial*. [En ligne]. Consulté à l'adresse : https://www.pythontutorial.net/tkinter/

[11]. GeeksForGeeks. *Setting the position of TKinter labels*. [En ligne]. Consulté à l'adresse : https://www.geeksforgeeks.org/setting-the-position-of-tkinter-labels/

[12]. MonPythonPasAPas. *tkinter.ttk.Combobox( )*. [En ligne]. Consulté à l'adresse : https://sites.google.com/site/pythonpasapas/modules/tkinter/tkinter-ttk/tkinter-ttk-combobox

[13]. CustomTkinter. *CustomTkinter - A modern and customizable python UI-library based on Tkinter*. [En ligne]. Consulté à l'adresse : https://customtkinter.tomschimansky.com (customtkinter, pas utilisé dans le rendu final du projet)