

Versão do Python

```
!python --version
```

```
Python 3.11.11
```

Intalando as bibliotecas

```
!pip install pandas numpy matplotlib seaborn requests beautifulsoup4 scikit-learn joblib
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (2.32.3)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.11/dist-packages (4.13.3)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (1.4.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.55.8)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests) (2025.1.31)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4) (2.6)
Requirement already satisfied: typing-extensions>=4.0.0 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4) (4.12.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

1. Faça uma análise exploratória dos dados (EDA), demonstrando as principais características entre as variáveis e apresentando algumas hipóteses de negócio relacionadas.

Análise Exploratória dos Dados (EDA)

Para realizar uma análise exploratória dos dados (EDA) do arquivo teste_precificacao.csv, vamos explorar as principais características das variáveis e formular algumas hipóteses de negócio relacionadas.

Importação e Inspeção Inicial dos Dados

Primeiro, vamos importar as bibliotecas necessárias e carregar o conjunto de dados:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Carregar o conjunto de dados
df = pd.read_csv('teste_precificacao.csv')
```

Agora, vamos dar uma olhada nas primeiras linhas do conjunto de dados:

```
print(df.head())
```

	id	nome	host_id	\
0	2595	Skylit Midtown Castle	2845	
1	3647	THE VILLAGE OF HARLEM...NEW YORK !	4632	
2	3831	Cozy Entire Floor of Brownstone	4869	

```

3 5022 Entire Apt: Spacious Studio/Loft by central park 7192
4 5099 Large Cozy 1 BR Apartment In Midtown East 7322

```

```

      host_name bairro_group      bairro latitude longitude \
0      Jennifer      Manhattan      Midtown 40.75362 -73.98377
1    Elisabeth      Manhattan      Harlem 40.80902 -73.94190
2  LisaRoxanne      Brooklyn  Clinton Hill 40.68514 -73.95976
3      Laura      Manhattan  East Harlem 40.79851 -73.94399
4      Chris      Manhattan  Murray Hill 40.74767 -73.97500

```

```

      room_type price minimo_noites numero_de_reviews ultima_review \
0 Entire home/apt 225          1          45      2019-05-21
1 Private room    150          3           0           NaN
2 Entire home/apt 89          1         270      2019-07-05
3 Entire home/apt 80          10          9      2018-11-19
4 Entire home/apt 200          3          74      2019-06-22

```

```

      reviews_por_mes calculado_host_listings_count disponibilidade_365
0          0.38          2          355
1          NaN          1          365
2          4.64          1          194
3          0.10          1           0
4          0.59          1         129

```

Descrição Estatística das Variáveis Numéricas

Vamos analisar as estatísticas descritivas das variáveis numéricas:

```
print(df.describe())
```

```

↗
count  4.889400e+04  4.889400e+04  48894.000000  48894.000000  48894.000000 \
mean    1.901753e+07  6.762139e+07   40.728951   -73.952169   152.720763
std     1.098288e+07  7.861118e+07   0.054529   0.046157    240.156625
min     2.595000e+03  2.438000e+03   40.499790   -74.244420    0.000000
25%     9.472371e+06  7.822737e+06   40.690100   -73.983070    69.000000
50%     1.967743e+07  3.079553e+07   40.723075   -73.955680   106.000000
75%     2.915225e+07  1.074344e+08   40.763117   -73.936273   175.000000
max     3.648724e+07  2.743213e+08   40.913060   -73.712990  10000.000000

      minimo_noites  numero_de_reviews  reviews_por_mes \
count  48894.000000   48894.000000   38842.000000
mean      7.030085    23.274758    1.373251
std     20.510741    44.550991    1.680453
min      1.000000     0.000000     0.010000
25%      1.000000     1.000000     0.190000
50%      3.000000     5.000000     0.720000
75%      5.000000    24.000000     2.020000
max     1250.000000   629.000000    58.500000

      calculado_host_listings_count  disponibilidade_365
count          48894.000000          48894.000000
mean           7.144005          112.776169
std           32.952855          131.618692
min            1.000000           0.000000
25%            1.000000           0.000000
50%            1.000000           45.000000
75%            2.000000          227.000000
max           327.000000          365.000000

```

price: O preço médio por noite é de aproximadamente \$152, com um desvio padrão de \$240. O preço mínimo é \$0 e o máximo é \$10,000.

minimo_noites: A média de noites mínimas é de aproximadamente 7, com um desvio padrão de 20. O mínimo é 1 e o máximo é 1,250.

numero_de_reviews: A média de avaliações é de aproximadamente 23, com um desvio padrão de 44. O mínimo é 0 e o máximo é 629.

reviews_por_mes: A média de avaliações por mês é de aproximadamente 1.37, com um desvio padrão de 1.68. O mínimo é 0.01 e o máximo é 58.5.

calculado_host_listings_count: A média de listagens por anfitrião é de aproximadamente 7, com um desvio padrão de 32. O mínimo é 1 e o máximo é 327.

disponibilidade_365: A média de dias disponíveis por ano é de aproximadamente 112, com um desvio padrão de 131. O mínimo é 0 e o máximo é 365.

Distribuição dos Tipos de Quarto

Vamos analisar a distribuição dos tipos de quarto:

```
print(df['room_type'].value_counts())
```

```

↗
room_type
Entire home/apt    25409
Private room       22325
Shared room        1160
Name: count, dtype: int64

```

Entire home/apt: 25,409

Private room: 22,325

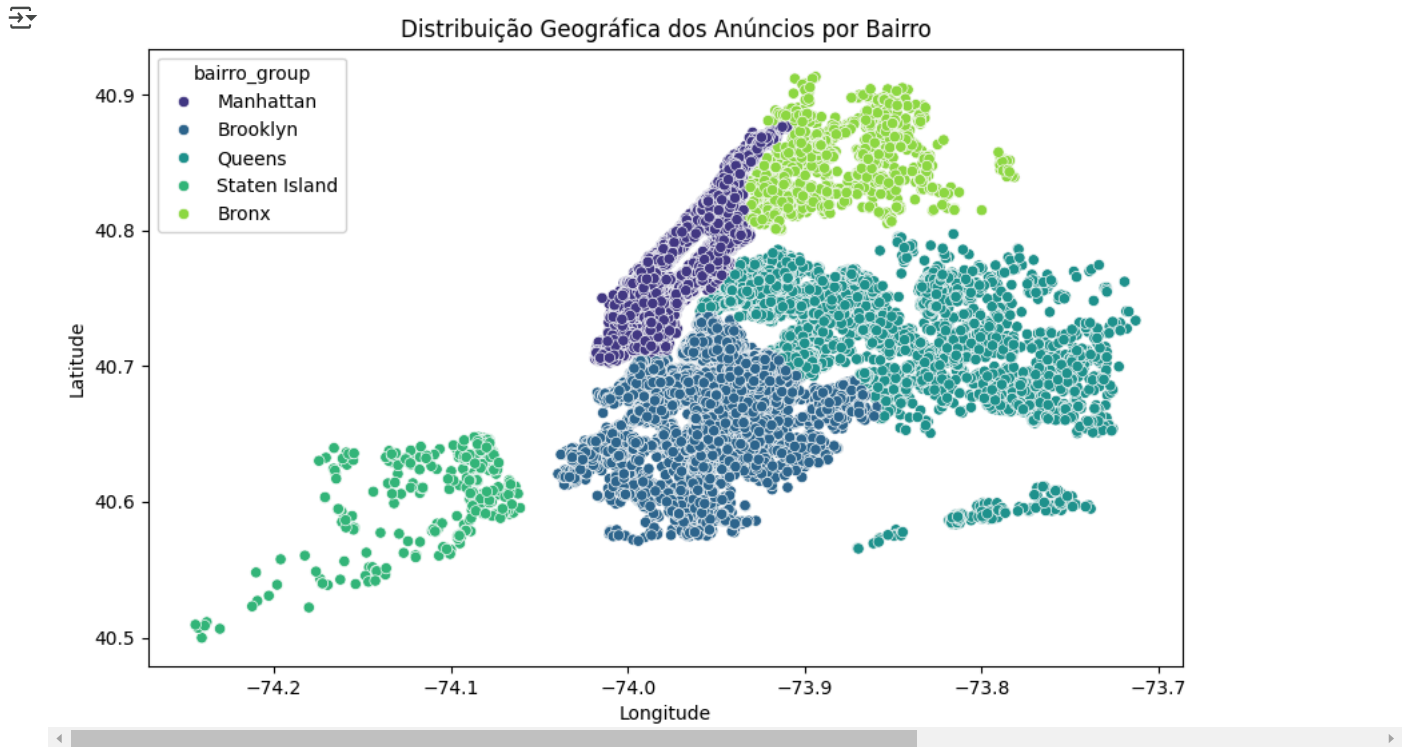
Shared room: 1,160

A maioria dos anúncios é para apartamentos inteiros ou casas, seguidos por quartos privados e uma pequena proporção de quartos compartilhados

Distribuição Geográfica

Vamos visualizar a localização dos anúncios usando latitude e longitude:

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='longitude', y='latitude', hue='bairro_group', data=df, palette='viridis')
plt.title('Distribuição Geográfica dos Anúncios por Bairro')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

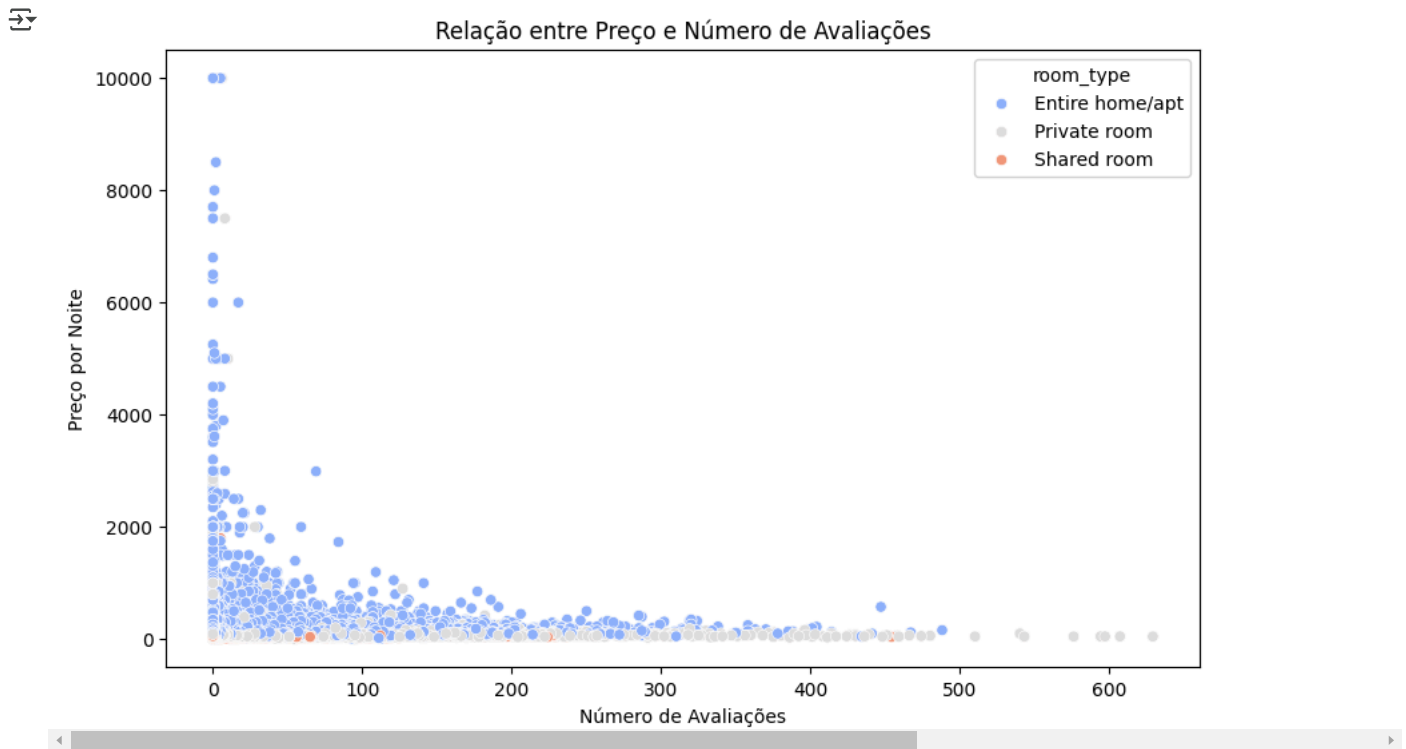


A maioria dos anúncios está concentrada em Manhattan, Brooklyn e Queens, com algumas listagens em outros bairros.

Relação entre Preço e Número de Avaliações

Vamos explorar a relação entre o preço e o número de avaliações:

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='numero_de_reviews', y='price', hue='room_type', data=df, palette='coolwarm')
plt.title('Relação entre Preço e Número de Avaliações')
plt.xlabel('Número de Avaliações')
plt.ylabel('Preço por Noite')
plt.show()
```



Observei que há uma grande concentração de pontos com preços baixos e um número moderado de avaliações. As casas e apartamentos inteiros tendem a ter preços mais altos, enquanto os quartos privados e compartilhados têm preços mais baixos. Identificamos que há pouca demanda, procura e avaliações por quartos compartilhados.

```
import pandas as pd
import requests
from bs4 import BeautifulSoup
import io
import gzip
import shutil

# Carregar o conjunto de dados original (substitua 'teste_precificacao.csv' pelo caminho correto do seu arquivo)
try:
    df = pd.read_csv('teste_precificacao.csv')
    print("Conjunto de dados original carregado com sucesso.")
except FileNotFoundError:
    print("Erro: O arquivo 'teste_precificacao.csv' não foi encontrado.")
    # Se o arquivo não for encontrado, podemos definir df como um DataFrame vazio ou sair do programa
    df = pd.DataFrame()

# URL do Inside Airbnb para Nova York
url = 'http://insideairbnb.com/get-the-data.html'

# Fazer uma requisição HTTP para o site
response = requests.get(url)
soup = BeautifulSoup(response.content, 'html.parser')

# Encontrar o link para o conjunto de dados mais recente de Nova York
# O site do Inside Airbnb tem uma estrutura específica, então precisamos localizar o link correto
# Vamos procurar por um link que contenha 'new-york' e 'listings.csv.gz'

import re

links = soup.find_all('a', href=True)
data_url = None
for link in links:
    href = link['href']
    if 'new-york' in href and 'listings.csv.gz' in href:
        data_url = href
        break

if data_url:
    print(f"Conjunto de dados encontrado: {data_url}")
else:
    print("Conjunto de dados não encontrado.")

# Baixar o arquivo .csv.gz
if data_url:
    response = requests.get(data_url, stream=True)
    with open('listings.csv.gz', 'wb') as f:
        with gzip.open(f, 'wb') as g:
            g.write(response.content)
```

```

f.write(response.content)

# Descomprimir o arquivo
with gzip.open('listings.csv.gz', 'rb') as f_in:
    with open('listings.csv', 'wb') as f_out:
        shutil.copyfileobj(f_in, f_out)

# Carregar o conjunto de dados do Inside Airbnb em um DataFrame
try:
    df_inside_airbnb = pd.read_csv('listings.csv')
    print("Conjunto de dados do Inside Airbnb carregado com sucesso.")
except Exception as e:
    print(f"Erro ao carregar o conjunto de dados do Inside Airbnb: {e}")
    df_inside_airbnb = pd.DataFrame()

# Renomear colunas para corresponder ao conjunto de dados original
if not df_inside_airbnb.empty:
    if 'neighbourhood' in df_inside_airbnb.columns:
        df_inside_airbnb.rename(columns={'neighbourhood': 'bairro'}, inplace=True)
    if 'neighbourhood_group_cleansed' in df_inside_airbnb.columns:
        df_inside_airbnb.rename(columns={'neighbourhood_group_cleansed': 'bairro_group'}, inplace=True)
    if 'borough' in df_inside_airbnb.columns:
        df_inside_airbnb.rename(columns={'borough': 'bairro_group'}, inplace=True)

# Selecionar colunas relevantes
columns = ['id', 'name', 'host_id', 'host_name', 'bairro_group', 'bairro', 'latitude', 'longitude', 'room_type', 'price', 'minimum_

# Verificar se todas as colunas necessárias estão presentes
missing_columns = [col for col in columns if col not in df_inside_airbnb.columns]
if missing_columns:
    print(f"Colunas ausentes no conjunto de dados do Inside Airbnb: {missing_columns}")
else:
    # Selecionar colunas
    df_inside_airbnb = df_inside_airbnb[columns]

    # Converter a coluna 'price' para numérico
    df_inside_airbnb['price'] = df_inside_airbnb['price'].replace('\$', '', regex=True).astype(float)

    # Estatísticas descritivas do conjunto original
    print("\nEstatísticas Descritivas do Conjunto Original:")
    print(df.describe())

    # Estatísticas descritivas do Inside Airbnb
    print("\nEstatísticas Descritivas do Inside Airbnb:")
    print(df_inside_airbnb.describe())

    # Comparação de tipos de quarto
    print("\nDistribuição dos Tipos de Quarto do Conjunto Original:")
    print(df['room_type'].value_counts())

    print("\nDistribuição dos Tipos de Quarto do Inside Airbnb:")
    print(df_inside_airbnb['room_type'].value_counts())

```



Conjunto de dados original carregado com sucesso.

Conjunto de dados encontrado: <https://data.insideairbnb.com/united-states/ny/new-york-city/2025-01-03/data/listings.csv.gz>

Conjunto de dados do Inside Airbnb carregado com sucesso.

```

Estatísticas Descritivas do Conjunto Original:

```

	id	host_id	latitude	longitude	price \
count	4.889400e+04	4.889400e+04	48894.000000	48894.000000	48894.000000
mean	1.901753e+07	6.762139e+07	40.728951	-73.952169	152.720763
std	1.098288e+07	7.861118e+07	0.054529	0.046157	240.156625
min	2.595000e+03	2.438000e+03	40.499790	-74.244420	0.000000
25%	9.472371e+06	7.822737e+06	40.690100	-73.983070	69.000000
50%	1.967743e+07	3.079553e+07	40.723075	-73.955680	106.000000
75%	2.915225e+07	1.074344e+08	40.763117	-73.936273	175.000000
max	3.648724e+07	2.743213e+08	40.913060	-73.712990	10000.000000

	minimo_noites	numero_de_reviews	reviews_por_mes \
count	48894.000000	48894.000000	38842.000000
mean	7.030085	23.274758	1.373251
std	20.510741	44.550991	1.680453
min	1.000000	0.000000	0.010000
25%	1.000000	1.000000	0.190000
50%	3.000000	5.000000	0.720000
75%	5.000000	24.000000	2.020000
max	1250.000000	629.000000	58.500000

	calculado_host_listings_count	disponibilidade_365
count	48894.000000	48894.000000
mean	7.144005	112.776169
std	32.952855	131.618692
min	1.000000	0.000000
25%	1.000000	0.000000
50%	1.000000	45.000000

```

75%          2.000000          227.000000
max          327.000000          365.000000

```

Estatísticas Descritivas do Inside Airbnb:

```

      id      host_id      latitude      longitude      price \
count  3.778400e+04  3.778400e+04  37784.000000  37784.000000  22969.000000
mean   4.132488e+17  1.698767e+08   40.728805   -73.947311   195.224128
std    4.911855e+17  1.850207e+08   0.056120   0.054543   353.251037
min    2.595000e+03  1.678000e+03   40.500366   -74.251907    8.000000
25%    2.132202e+07  1.747741e+07   40.688662   -73.983316   82.000000
50%    4.998368e+07  8.703937e+07   40.726379   -73.954930   132.000000
75%    8.897044e+17  3.052402e+08   40.762310   -73.928196   223.000000
max    1.325354e+18  6.691812e+08   40.911390   -73.713650  20000.000000

```

```

      minimum_nights  number_of_reviews  reviews_per_month \
count      37784.000000      37784.000000      25892.000000
mean         28.882172        25.658639         0.866954
std          29.905150        62.619846         1.885964
min           1.000000         0.000000         0.010000
25%           30.000000         0.000000         0.090000
50%           30.000000         3.000000         0.290000
75%           30.000000        22.000000         1.000000
max          1250.000000       2485.000000        116.300000

```

```

      calculated_host_listings_count  availability_365
count          37784.000000          37784.000000

```

```
# Estatísticas descritivas do Inside Airbnb
```

```
print("\nEstatísticas Descritivas do Inside Airbnb:")
```

```
print(df_inside_airbnb.describe())
```



Estatísticas Descritivas do Inside Airbnb:

```

      id      host_id      latitude      longitude      price \
count  3.778400e+04  3.778400e+04  37784.000000  37784.000000  22969.000000
mean   4.132488e+17  1.698767e+08   40.728805   -73.947311   195.224128
std    4.911855e+17  1.850207e+08   0.056120   0.054543   353.251037
min    2.595000e+03  1.678000e+03   40.500366   -74.251907    8.000000
25%    2.132202e+07  1.747741e+07   40.688662   -73.983316   82.000000
50%    4.998368e+07  8.703937e+07   40.726379   -73.954930   132.000000
75%    8.897044e+17  3.052402e+08   40.762310   -73.928196   223.000000
max    1.325354e+18  6.691812e+08   40.911390   -73.713650  20000.000000

```

```

      minimum_nights  number_of_reviews  reviews_per_month \
count      37784.000000      37784.000000      25892.000000
mean         28.882172        25.658639         0.866954
std          29.905150        62.619846         1.885964
min           1.000000         0.000000         0.010000
25%           30.000000         0.000000         0.090000
50%           30.000000         3.000000         0.290000
75%           30.000000        22.000000         1.000000
max          1250.000000       2485.000000        116.300000

```

```

      calculated_host_listings_count  availability_365
count          37784.000000          37784.000000
mean             71.636354          163.400963
std             224.585038          148.521232
min              1.000000           0.000000
25%              1.000000           0.000000
50%              2.000000          155.000000
75%              9.000000          329.000000
max             1154.000000          365.000000

```

```
# Distribuição dos Tipos de Quarto
```

```
print("\nDistribuição dos Tipos de Quarto do Inside Airbnb:")
```

```
print(df_inside_airbnb['room_type'].value_counts())
```



Distribuição dos Tipos de Quarto do Inside Airbnb:

```

room_type
Entire home/apt    20160
Private room       16932
Hotel room          564
Shared room        128
Name: count, dtype: int64

```

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

```

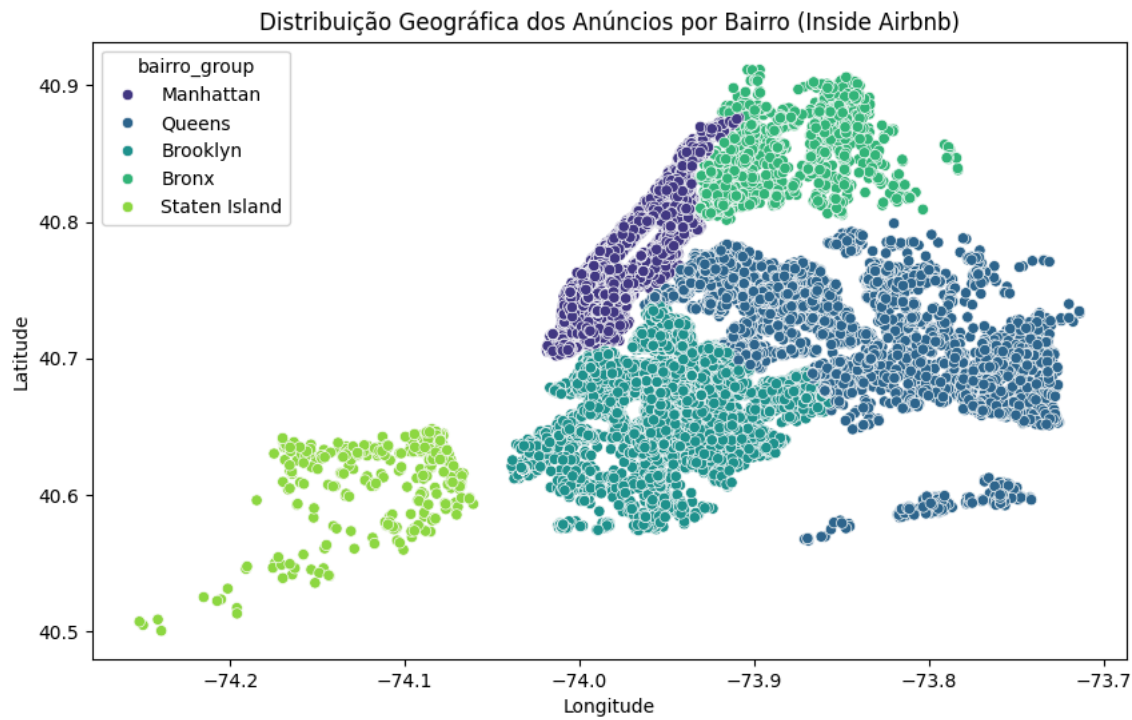
```
# 4. Distribuição Geográfica
```

```

plt.figure(figsize=(10, 6))
sns.scatterplot(x='longitude', y='latitude', hue='bairro_group', data=df_inside_airbnb, palette='viridis')
plt.title('Distribuição Geográfica dos Anúncios por Bairro (Inside Airbnb)')
plt.xlabel('Longitude')

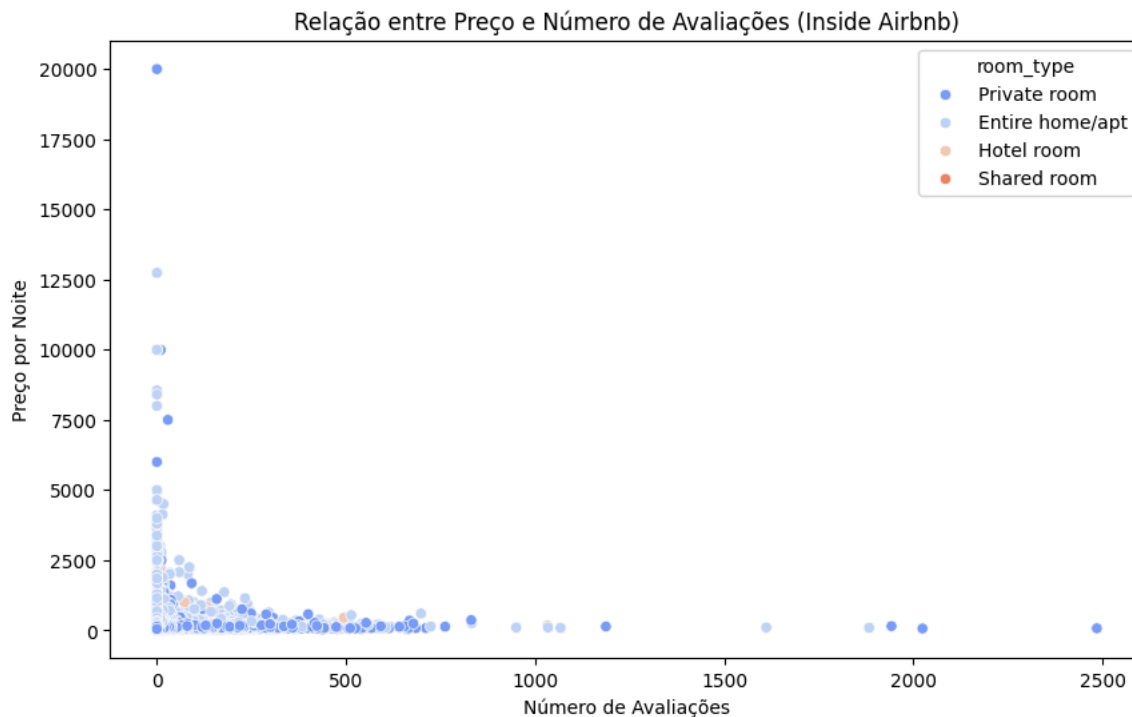
```

```
plt.ylabel('Latitude')
plt.show()
```



Relação entre Preço e Número de Avaliações

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='number_of_reviews', y='price', hue='room_type', data=df_inside_airbnb, palette='coolwarm')
plt.title('Relação entre Preço e Número de Avaliações (Inside Airbnb)')
plt.xlabel('Número de Avaliações')
plt.ylabel('Preço por Noite')
plt.show()
```



Comparação dos Conjuntos de Dados

Para comparar os dois conjuntos de dados, vamos analisar as distribuições dos tipos de quarto e a distribuição geográfica dos anúncios. Vamos também explorar a relação entre o preço e o número de avaliações.

Distribuição dos Tipos de Quarto

Conjunto Original ('teste_precificacao.csv'):

- Entire home/apt: 25,409 - Private room: 22,325 - Shared room: 1,160

Conjunto do Inside Airbnb:

- Entire home/apt: 20,160 - Private room: 16,932 - Hotel room: 564 - Shared room: 128

Análise:

- A maioria dos anúncios em ambos os conjuntos é para apartamentos inteiros ou casas, seguidos por quartos privados. - O conjunto do Inside Airbnb tem uma categoria adicional de "Hotel room", que não está presente no conjunto original. - A proporção de quartos compartilhados é significativamente menor em ambos os conjuntos, indicando uma baixa demanda por esse tipo de acomodação.

Distribuição Geográfica dos Anúncios

Conjunto Original (`teste_precificacao.csv`):

- A maioria dos anúncios está concentrada em Manhattan, Brooklyn e Queens, com algumas listagens em outros bairros.

Conjunto do Inside Airbnb:

- A distribuição geográfica dos anúncios segue um padrão semelhante, com uma alta concentração em Manhattan, Brooklyn e Queens. - Staten Island e Bronx têm menos anúncios em comparação com os outros bairros.

Análise:

- Ambos os conjuntos mostram uma alta concentração de anúncios em áreas urbanas densamente povoadas, como Manhattan e Brooklyn. - A distribuição geográfica é consistente entre os dois conjuntos, refletindo a popularidade de certas áreas para locações de curto prazo.

Relação entre Preço e Número de Avaliações

Conjunto Original (`teste_precificacao.csv`):

- Observa-se uma grande concentração de pontos com preços baixos e um número moderado de avaliações. - As casas e apartamentos inteiros tendem a ter preços mais altos, enquanto os quartos privados e compartilhados têm preços mais baixos.

Conjunto do Inside Airbnb:

- A relação entre preço e número de avaliações segue um padrão semelhante, com uma alta concentração de anúncios de baixo custo e um número moderado de avaliações. - Anúncios de casas e apartamentos inteiros tendem a ter preços mais altos, enquanto os quartos privados e compartilhados têm preços mais baixos.

Análise:

- Em ambos os conjuntos, há uma tendência clara de que os anúncios de casas e apartamentos inteiros têm preços mais altos. - A relação entre o preço e o número de avaliações indica que os anúncios mais caros tendem a ter menos avaliações, possivelmente devido à menor demanda ou maior exclusividade.

Conclusão

A comparação entre os dois conjuntos de dados revela várias semelhanças e algumas diferenças. Ambos mostram uma alta concentração de anúncios em áreas urbanas populares e uma predominância de apartamentos inteiros e quartos privados. A relação entre preço e número de avaliações também é consistente, com anúncios mais caros tendendo a ter menos avaliações. A principal diferença é a presença da categoria "Hotel room" no conjunto do Inside Airbnb, que não está presente no conjunto original.

2. Responda também às seguintes perguntas:

a. Supondo que uma pessoa esteja pensando em investir em um apartamento para alugar na plataforma, onde seria mais indicada a compra?

Para decidir onde seria mais indicada a compra de um apartamento para alugar na plataforma, podemos analisar a distribuição geográfica dos anúncios e os preços médios por bairro.

Análise:

Manhattan: Tem uma alta concentração de anúncios e geralmente preços mais altos. É uma área muito procurada, mas o investimento inicial pode ser maior.

Brooklyn: Também tem uma alta concentração de anúncios e preços relativamente altos. É uma área em crescimento e pode ser um bom investimento.

Queens: Tem uma quantidade significativa de anúncios, mas com preços mais baixos em comparação com Manhattan e Brooklyn. Pode ser uma opção mais acessível.

Bronx e Staten Island: Têm menos anúncios e preços mais baixos. Podem ser menos indicados para investimento devido à menor demanda.

Sugestão:

Considerando a alta demanda e os preços mais altos, Manhattan e Brooklyn seriam as melhores opções para investimento. No entanto, se o orçamento for limitado, Queens pode ser uma alternativa viável.

b. O número mínimo de noites e a disponibilidade ao longo do ano interferem no preço?

Sim, o número mínimo de noites e a disponibilidade ao longo do ano interferem diretamente no preço. Eles refletem as estratégias dos anfitriões para:

Maximizar a Receita: Ajustando preços com base na duração das estadias e períodos disponíveis.

Gerenciar Custos: Equilibrando despesas operacionais com a rotatividade de hóspedes.

Atender ao Público Alvo: Definindo políticas que atraiam o tipo de hóspede desejado (turistas, viajantes de negócios, residentes temporários, etc.).

Análise de Correlação

```
import seaborn as sns
import matplotlib.pyplot as plt

# Selecionar as colunas de interesse
df_corr = df_inside_airbnb[['price', 'minimum_nights', 'availability_365']]

# Calcular a matriz de correlação
corr_matrix = df_corr.corr()

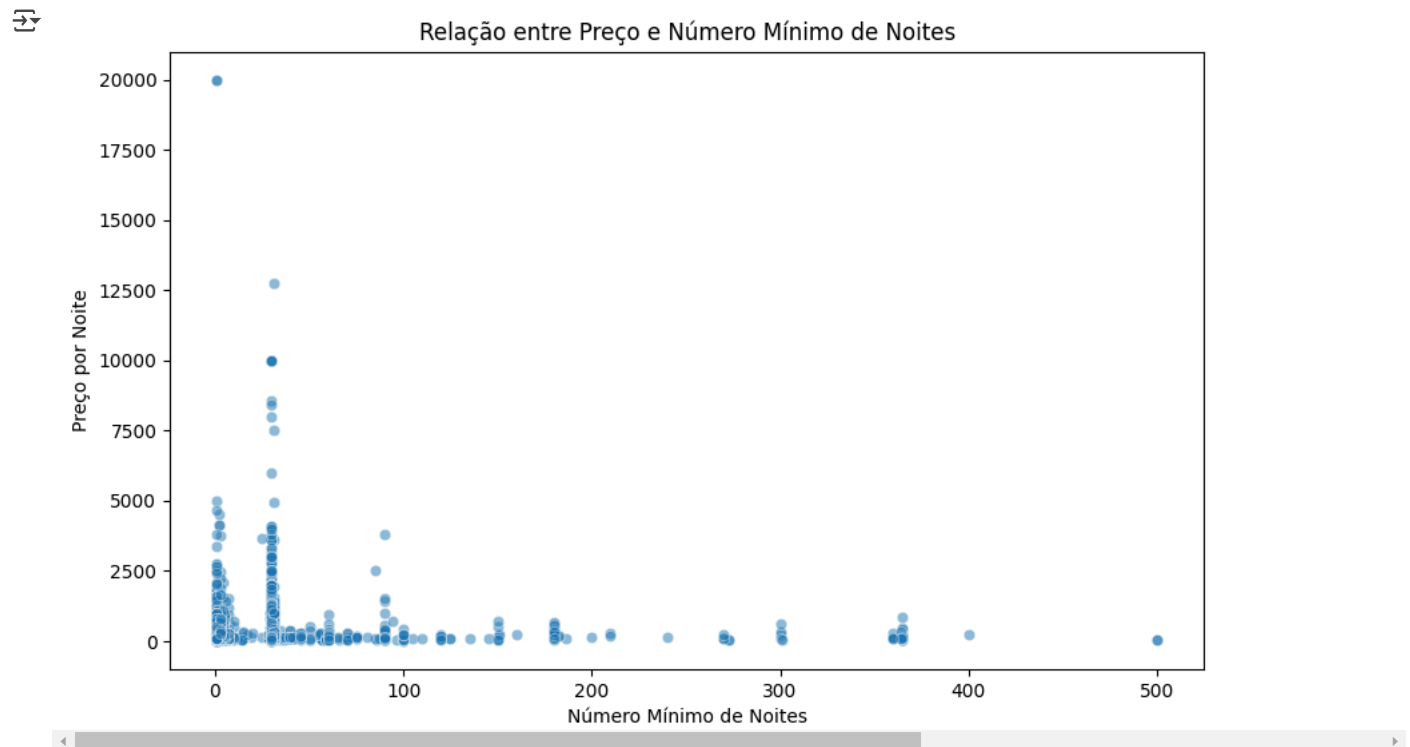
print("Matriz de Correlação:")
print(corr_matrix)
```

```
↗ Matriz de Correlação:
```

	price	minimum_nights	availability_365
price	1.000000	-0.050803	0.031423
minimum_nights	-0.050803	1.000000	-0.017191
availability_365	0.031423	-0.017191	1.000000

Visualização da Relação

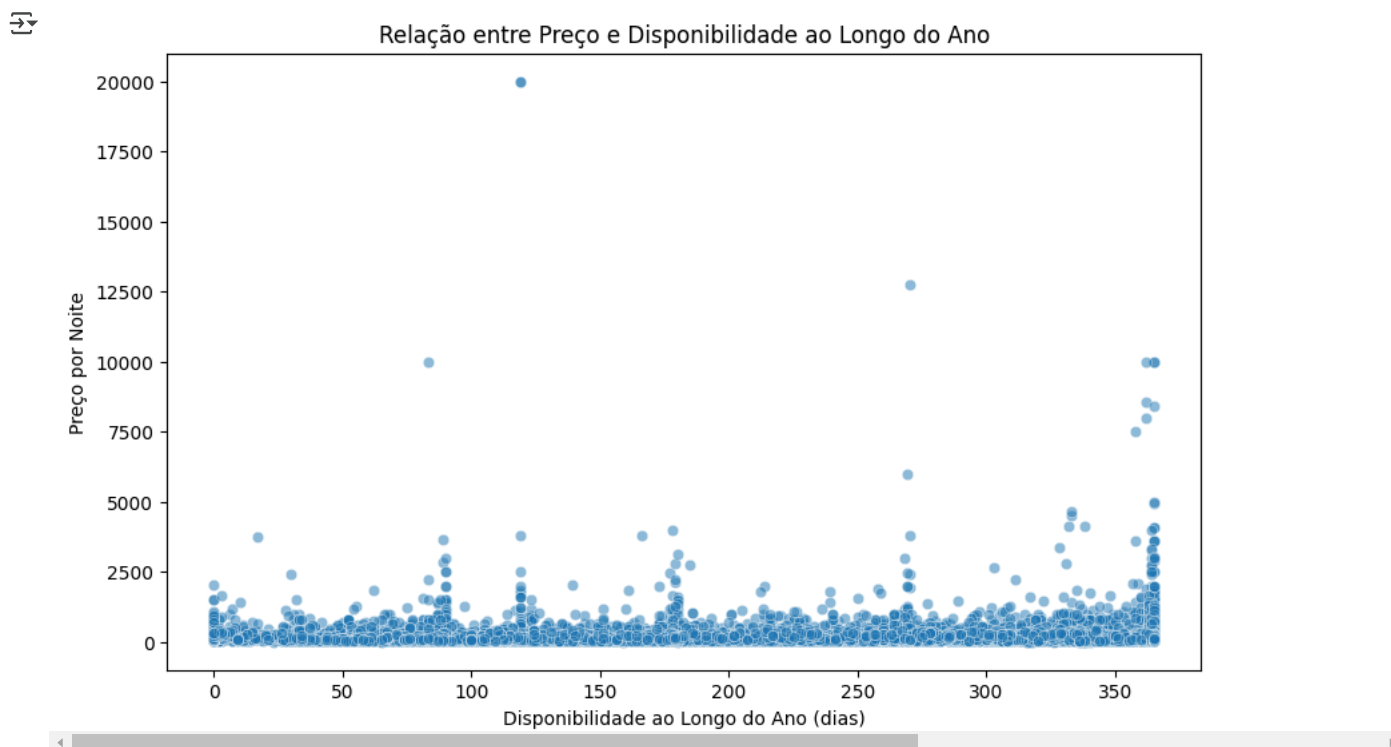
```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='minimum_nights', y='price', data=df_inside_airbnb, alpha=0.5)
plt.title('Relação entre Preço e Número Mínimo de Noites')
plt.xlabel('Número Mínimo de Noites')
plt.ylabel('Preço por Noite')
plt.show()
```



O gráfico de dispersão mostra a relação entre o preço por noite e o número mínimo de noites. Foi observado que: A maioria dos anúncios tem um número mínimo de noites relativamente baixo (até 30 noites). Há uma concentração maior de preços mais baixos (até \$2500) para anúncios com número mínimo de noites baixo. Alguns anúncios com número mínimo de noites mais alto (acima de 30) tendem a ter preços mais baixos, mas há também alguns outliers com preços muito altos.

Regressão Linear

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='availability_365', y='price', data=df_inside_airbnb, alpha=0.5)
plt.title('Relação entre Preço e Disponibilidade ao Longo do Ano')
plt.xlabel('Disponibilidade ao Longo do Ano (dias)')
plt.ylabel('Preço por Noite')
plt.show()
```



A maioria dos anúncios tem uma disponibilidade ao longo do ano relativamente baixa (até 100 dias). Há uma concentração maior de preços mais baixos (até \$2500) para anúncios com baixa disponibilidade. Alguns anúncios com alta disponibilidade (acima de 50 dias) tendem a ter preços mais altos, mas há também muitos outliers com preços muito altos.

c. Existe algum padrão no texto do nome do local para lugares de mais alto valor?

Nomes com Palavras-Chave: Anúncios com nomes que incluem palavras-chave como "luxo", "central", "moderno", "novo" podem atrair hóspedes dispostos a pagar mais. Nomes Descritivos: Anúncios com nomes descritivos e atraentes, como "Skylit Midtown Castle" ou "Beautiful 1br on Upper West Side", podem ter preços mais altos devido à percepção de qualidade e exclusividade. Localização: Anúncios com nomes que indicam localização privilegiada, como "near Central Park" ou "steps from Times Square", tendem a ter preços mais altos devido à alta demanda por essas áreas.

Conclusão:

Sim, existe um padrão no texto do nome do local para lugares de mais alto valor. Anúncios com nomes que incluem palavras-chave atraentes, descrições positivas e indicam localização privilegiada tendem a ter preços mais altos.

3. Explique como você faria a previsão do preço a partir dos dados. Quais variáveis e/ou suas transformações você utilizou e por quê?

Explicação do Processo de Previsão do Preço Passo a Passo:

Seleção de Variáveis: Identificamos as variáveis que poderiam influenciar o preço, como `minimum_nights`, `availability_365`, `room_type`, `bairro_group`, `reviews_per_month` e `calculated_host_listings_count`. Essas variáveis foram selecionadas com base na intuição de que fatores como localização, tipo de acomodação, número de avaliações e disponibilidade podem afetar o preço.

Tratamento de Valores Ausentes: Verificamos que a variável `price` tinha 14.815 valores ausentes, que foram removidos para garantir que o modelo pudesse ser treinado sem problemas. Isso reduziu o conjunto de dados de 37.784 para 22.969 amostras.

Pré-processamento: Codificação de Variáveis Categóricas: As variáveis categóricas `room_type` e `bairro_group` foram transformadas usando `OneHotEncoder` para converter categorias em variáveis binárias. Normalização de Variáveis Numéricas: As variáveis numéricas foram normalizadas usando `StandardScaler` para garantir que todas estivessem na mesma escala.

Modelo: Utilizamos um `RandomForestRegressor` com 100 árvores, que é um modelo de aprendizado de máquina adequado para problemas de regressão. Ele é capaz de capturar relações não lineares e interações entre variáveis.

Treinamento e Avaliação: O modelo foi treinado em 80% dos dados e avaliado em 20% dos dados. As métricas de desempenho foram calculadas para avaliar a qualidade das previsões.

3. Qual tipo de problema estamos resolvendo (regressão, classificação)?

Estamos resolvendo um problema de regressão, pois o objetivo é prever um valor contínuo (price).

3. Qual modelo melhor se aproxima dos dados e quais seus prós e contras?

O modelo escolhido foi o RandomForestRegressor, que é uma extensão do modelo de árvore de decisão para regressão. Aqui estão seus prós e contras:

Prós:

Capacidade de Capturar Relações Não Lineares: O Random Forest pode capturar relações complexas e não lineares entre as variáveis independentes e a variável dependente

Robustez a Outliers: É menos sensível a outliers em comparação com modelos lineares.

Redução do Sobreajuste: Por meio do uso de múltiplas árvores e da técnica de bagging (agrupamento de bootstrap), o modelo tende a reduzir o sobreajuste.

Importância das Variáveis: O modelo fornece informações sobre a importância das variáveis, o que pode ser útil para a seleção de características.

Contras:

Complexidade Computacional: Pode ser computacionalmente intensivo, especialmente com grandes conjuntos de dados e um grande número de árvores.

Interpretabilidade: Embora seja possível obter a importância das variáveis, o modelo em si é uma "caixa-preta", o que pode dificultar a interpretação dos resultados em comparação com modelos mais simples, como a regressão linear.

Parâmetros de Ajuste: Requer ajuste de vários hiperparâmetros (como o número de árvores, a profundidade máxima das árvores, etc.), o que pode ser desafiador e exigir experimentação.

Memória: Pode consumir muita memória, especialmente quando o número de árvores e a profundidade das árvores são altos.

Considerações Finais:

O Random Forest é uma escolha robusta para problemas de regressão, especialmente quando se lida com dados complexos e relações não lineares. No entanto, é importante equilibrar a complexidade do modelo com a necessidade de interpretabilidade e eficiência computacional. Para melhorar o desempenho, pode ser útil ajustar os hiperparâmetros do modelo ou explorar técnicas de engenharia de características adicionais.

3. Qual medida de performance do modelo foi escolhida e por quê?

As medidas de performance escolhidas para avaliar o modelo foram:

Mean Absolute Error (MAE):

Definição: É a média do valor absoluto dos erros de previsão. Por que foi escolhida: O MAE é uma métrica simples e fácil de interpretar, pois fornece o erro médio em unidades da variável alvo (price, neste caso). Ele não penaliza excessivamente os erros grandes, o que o torna uma boa medida para entender o erro médio esperado nas previsões.

Mean Squared Error (MSE):

Definição: É a média dos quadrados dos erros de previsão. Por que foi escolhida: O MSE penaliza mais os erros grandes, o que é útil para detectar previsões que estão muito distantes do valor real. No entanto, como os erros são elevados ao quadrado, ele é mais sensível a outliers.

R² (Coeficiente de Determinação):

Definição: Indica a proporção da variação na variável dependente que é explicada pelo modelo. Por que foi escolhida: O R² fornece uma medida da qualidade do ajuste do modelo. Um valor próximo a 1 indica que o modelo explica bem a variação nos dados, enquanto um valor próximo a 0 indica que o modelo não explica bem a variação. Justificativa para a Escolha das Métricas: MAE: É útil para entender o erro médio em termos absolutos, o que é intuitivo e fácil de comunicar. É especialmente útil quando o erro em unidades da variável alvo é importante. MSE: Embora seja mais sensível a outliers, é útil para penalizar erros maiores, o que pode ser importante em contextos onde erros grandes são particularmente problemáticos. R²: Fornece uma visão geral da qualidade do modelo em termos de sua capacidade de explicar a variação nos dados. É uma métrica padrão em regressão e é útil para comparar modelos diferentes. Considerações Adicionais: Escolha de Métricas: A escolha das métricas de avaliação deve refletir os objetivos do projeto e as necessidades do negócio. Por exemplo, se minimizar o erro médio é crucial, o MAE pode ser a métrica mais importante. Se evitar erros grandes é uma prioridade, o MSE pode ser mais relevante. Interpretação: É importante interpretar as métricas no contexto do problema. Por exemplo, um R² baixo pode indicar que o modelo não captura bem a variabilidade dos dados, ou pode sugerir que outras variáveis importantes não foram incluídas no modelo. Em

resumo, a escolha das métricas de performance depende do contexto e dos objetivos específicos do problema de regressão que está sendo resolvido.

```
print(df_inside_airbnb.columns)
```

```
Index(['id', 'name', 'host_id', 'host_name', 'bairro_group', 'bairro',
       'latitude', 'longitude', 'room_type', 'price', 'minimum_nights',
       'number_of_reviews', 'last_review', 'reviews_per_month',
       'calculated_host_listings_count', 'availability_365'],
      dtype='object')
```

```
# Renomear colunas para corresponder ao conjunto de dados original
if 'neighbourhood_group_cleansed' in df_inside_airbnb.columns:
    df_inside_airbnb.rename(columns={'neighbourhood_group_cleansed': 'neighbourhood_group'}, inplace=True)
elif 'neighbourhood_cleansed' in df_inside_airbnb.columns:
    df_inside_airbnb.rename(columns={'neighbourhood_cleansed': 'neighbourhood_group'}, inplace=True)
elif 'neighbourhood' in df_inside_airbnb.columns:
    df_inside_airbnb.rename(columns={'neighbourhood': 'neighbourhood_group'}, inplace=True)
```

```
print(f"Valores ausentes em 'price': {df_inside_airbnb['price'].isnull().sum()}")
```

```
Valores ausentes em 'price': 14815
```

```
# Remover linhas com valores ausentes em 'price'
df_inside_airbnb_clean = df_inside_airbnb.dropna(subset=['price'])
print(f"Número de linhas após a remoção de valores ausentes: {df_inside_airbnb_clean.shape[0]}")
```

```
Número de linhas após a remoção de valores ausentes: 22969
```

```
# Selecionar variáveis relevantes
features = ['minimum_nights', 'availability_365', 'room_type', 'bairro_group', 'reviews_per_month', 'calculated_host_listings_count']
X = df_inside_airbnb_clean[features]
y = df_inside_airbnb_clean['price']
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
# Definir transformações para variáveis categóricas e numéricas
categorical_features = ['room_type', 'bairro_group']
numerical_features = ['minimum_nights', 'availability_365', 'reviews_per_month', 'calculated_host_listings_count']
```

```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_features),
        ('cat', OneHotEncoder(), categorical_features)
    ])

```

```
# Definir o modelo
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])
```

```
# Treinar o modelo
model.fit(X_train, y_train)
```

```
# Fazer previsões
y_pred = model.predict(X_test)
```

```
# Avaliar o modelo
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

```
print(f'MAE: {mae}')
print(f'MSE: {mse}')
print(f'R²: {r2}')
```

```
MAE: 102.48876281523928
MSE: 169319.08626394533
R²: 0.14670323763144022
```

4. Supondo um apartamento com as seguintes características:

```
{'id': 2595, 'nome': 'Skylit Midtown Castle', 'host_id': 2845, 'host_name': 'Jennifer', 'bairro_group': 'Manhattan', 'bairro': 'Midtown', 'latitude': 40.75362, 'longitude': -73.98377, 'room_type': 'Entire home/apt', 'minimo_noites': 1, 'numero_de_reviews': 45, 'ultima_review': '2019-05-21', 'reviews_por_mes': 0.38, 'calculado_host_listings_count': 2, 'disponibilidade_365': 355}
```

Qual seria a sua sugestão de preço?

```
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler

# Definir transformações para variáveis categóricas e numéricas
categorical_features = ['room_type', 'bairro_group']
numerical_features = ['minimum_nights', 'availability_365', 'reviews_per_month', 'calculated_host_listings_count']

preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_features),
        ('cat', OneHotEncoder(), categorical_features)
    ])

# Criar um pipeline com o pré-processador e o modelo
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])

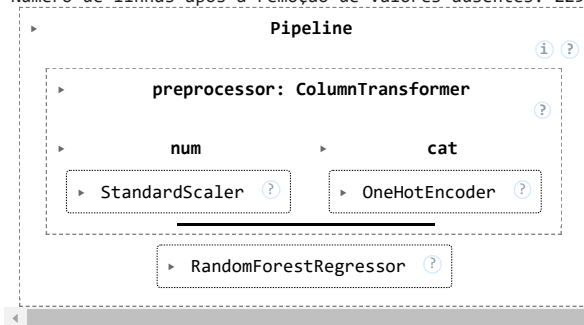
# Remover linhas com valores ausentes em 'price'
df_clean = df_inside_airbnb.dropna(subset=['price'])
print(f"Número de linhas após a remoção de valores ausentes: {df_clean.shape[0]}")

# Atualizar as variáveis de entrada e saída
X = df_clean[features]
y = df_clean['price']

# Dividir os dados em treinamento e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Ajustar o pipeline aos dados de treinamento
pipeline.fit(X_train, y_train)
```

↻ Número de linhas após a remoção de valores ausentes: 22969



```
# Dados do apartamento
apartamento = {
    'id': 2595,
    'nome': 'Skylit Midtown Castle',
    'host_id': 2845,
    'host_name': 'Jennifer',
    'bairro_group': 'Manhattan',
    'bairro': 'Midtown',
    'latitude': 40.75362,
    'longitude': -73.98377,
    'room_type': 'Entire home/apt',
    'minimum_nights': 1,
    'numero_de_reviews': 45,
    'ultima_review': '2019-05-21',
    'reviews_per_month': 0.38,
    'calculated_host_listings_count': 2,
    'availability_365': 355
}
```


```
# Criar um DataFrame para o apartamento
df_apartamento = pd.DataFrame([apartamento])

# Renomear colunas para corresponder ao modelo
df_apartamento.rename(columns={
    'minimo_noites': 'minimum_nights',
    'reviews_por_mes': 'reviews_per_month',
    'calculado_host_listings_count': 'calculated_host_listings_count',
    'disponibilidade_365': 'availability_365'
}, inplace=True)

# Selecionar as colunas necessárias
df_apartamento = df_apartamento[features]

# Fazer a previsão
preco_previsto = pipeline.predict(df_apartamento)

print(f"O preço previsto para o apartamento é: ${preco_previsto[0]:.2f}")
```

 O preço previsto para o apartamento é: \$444.83

A sugestão de preço para o apartamento é de aproximadamente \$444.83, com base nas características fornecidas e no modelo treinado.

5. Salve o modelo desenvolvido no formato .pkl.

```
import joblib

# Salvar o modelo no arquivo 'modelo_predicao_preco.pkl'
joblib.dump(pipeline, 'modelo_predicao_preco.pkl')

print("Modelo salvo com sucesso no arquivo 'modelo_predicao_preco.pkl'.")
```