Module_7_Assignment_Data_Visualization_Strategies

August 11, 2024

0.1 Dempsey Wade

1 Module 7 - Data visualization strategies

Author: Favio Vázquez

In this assigment, you will be creating some basic plots and also analyzing data from Airbnb. This dataset describes the listing activity and metrics in NYC, NY for 2019.

You should use the different plots and strategies for Exploratory Data Analysis that we've covered in the coding demos and videos with the professor.Please review the Important Instructions section below. You must adhere to these instructions to ensure the grading for this assignment works properly in Vocareum.

This assignment is designed to build your familiarity and comfort coding in Python while also helping you review key topics from each module. As you progress through the assignment, answers will get increasingly complex. It is important that you adopt a data scientist's mindset when completing this assignment. Remember to run your code from each cell before submitting your assignment. Running your code beforehand will notify you of errors and give you a chance to fix your errors before submitting. You should view your Vocareum submission as if you are delivering a final project to your manager or client.

1.0.1 IMPORTANT INSTRUCTIONS:

- Use `Seaborn` or `matplotlib` to solve every question.
- To be able to test for this module, you will be asked to save your figures as PNG into a folder called "results". Please don't change the name we ask you to give to the plots so you are able to get all the points in every question. The code you will use to save the PNG files is: plt.savefig("results/plot.png")
- Don't add any customization you're not asked to in the plots.

1.0.2 Index:

Basic plots:

** Airbnb plots**:

[1]:
```
# Import libraries. This cell must run.
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import numpy as np
import math

import warnings
warnings.filterwarnings('ignore')
```

1.1 Basic plots

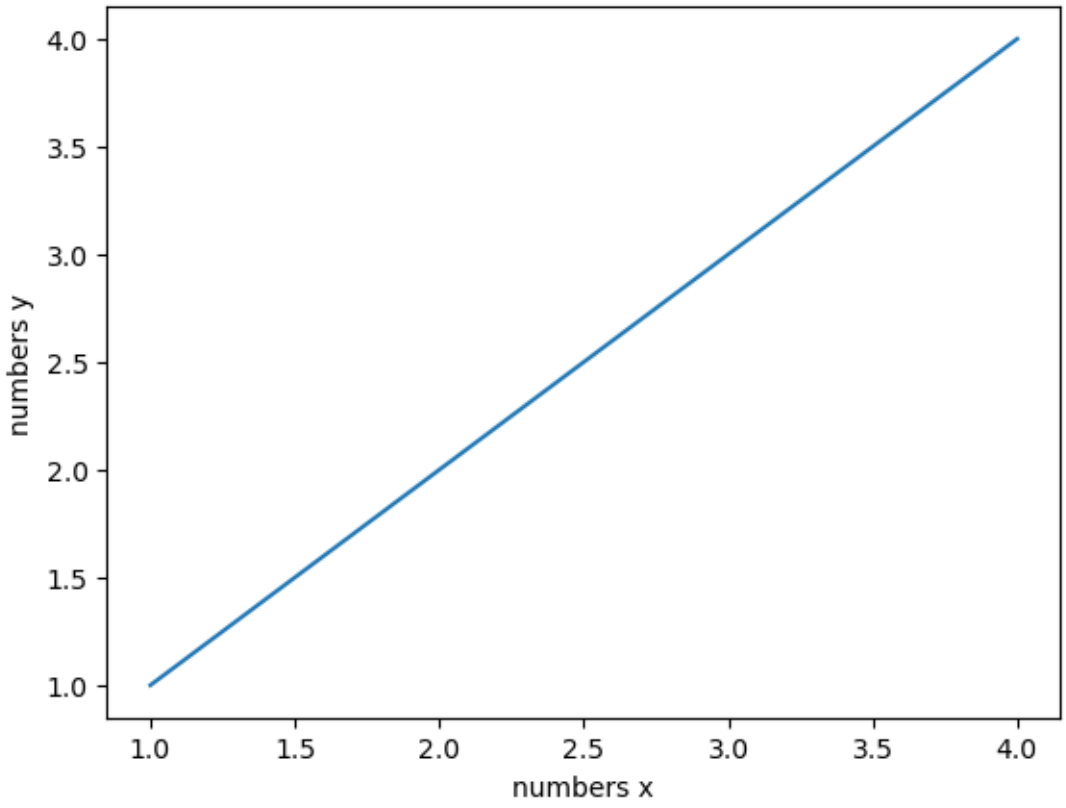
1.1.1 Question 0

This is a test question to get you familiarized with the grading in this assignment. Bellow we will use the command `plt.plot()` to create a basic lineplot with the data `x = [1,2,3,4]` and `y = [1,2,3,4]`. We will add a y-label with the words "numbers y" and add an x-label with the words "numbers x".

FInally we will use the pyplot function savefig to save your plot as a png file with the name "plot0.png" in the folder "results".

You don't have to change anything to get points from this question, just understand the structure of the plot creation and saving.

[2]:
```
### GRADED

# Data
x = [1,2,3,4]
y = [1,2,3,4]

### YOUR SOLUTION HERE
plt.plot([1,2,3,4],[1,2,3,4])
plt.ylabel("numbers y")
plt.xlabel("numbers x")
plt.savefig("results/plot0.png")
#plt.close()

###
### YOUR CODE HERE
###
```



[3]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.1.2 Question 1

In the code cell below, we have defined an x array with values from 0 to 2π.

Set y_sin equal to

$$\sin(x),$$

and y_cos equal to

$$3\cos(x).$$

On the same figure, plot `x` with `y_sin` and `x` with `y_cos` as a line plot.

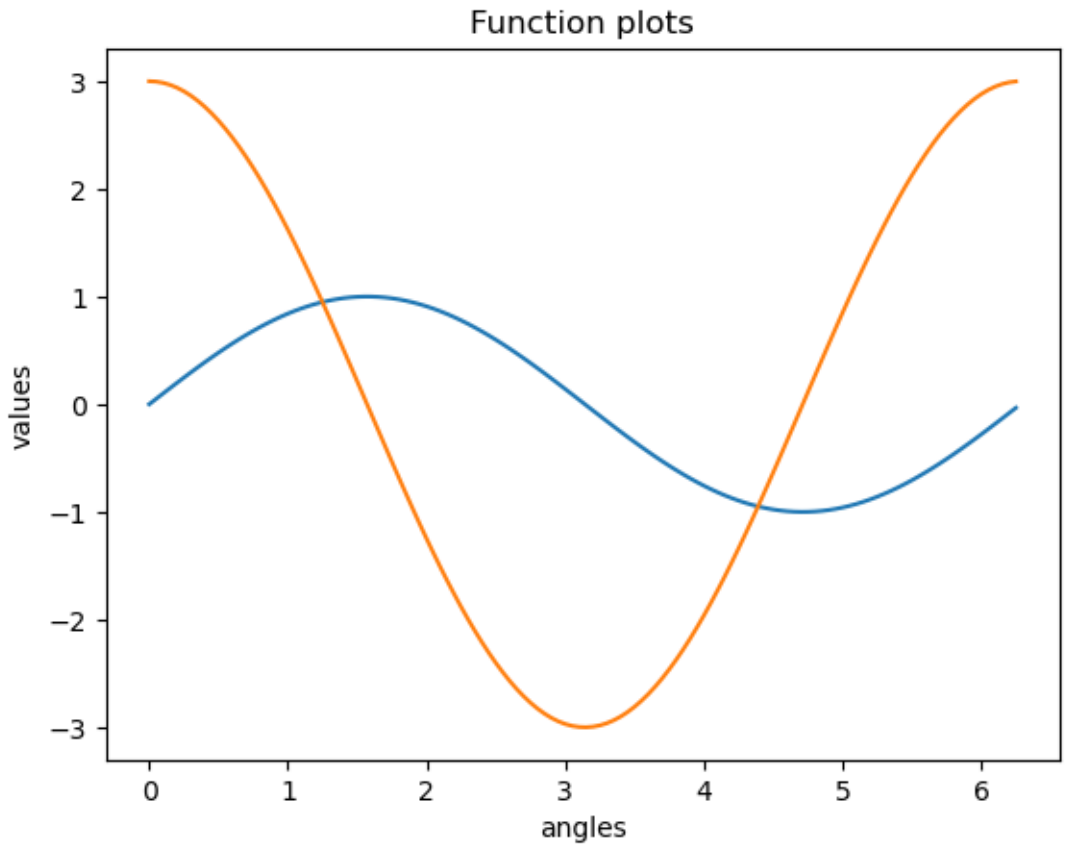
Customize the plot with the following settings:

- Title: "Function plots". **Hint: Notice the capitalization.**
- y-label = "values"
- x-label = "angles"

Save your plot as a png file with the name "plot1.png" in the folder "results".

HINT: Use the NumPy function sin and cos to define your dependent variables.

[4]:
```
### GRADED

import math
# the x axis: ndarray object of angles between 0 and 2
x = np.arange(0, math.pi*2, 0.05)
y_sin = [0]*len(x)
y_cos = [0]*len(x)

count = 0
for i in x:
    y_sin[count] = math.sin(i)
    y_cos[count] = 3*math.cos(i)
    count = count+1

plt.plot(x, y_sin)
plt.plot(x, y_cos)
plt.title("Function plots")
plt.ylabel("values")
plt.xlabel("angles")
plt.savefig("results/plot1.png")
plt.show()

### YOUR SOLUTION HERE

###
### YOUR CODE HERE
###
```



[5]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.1.3 Question 2

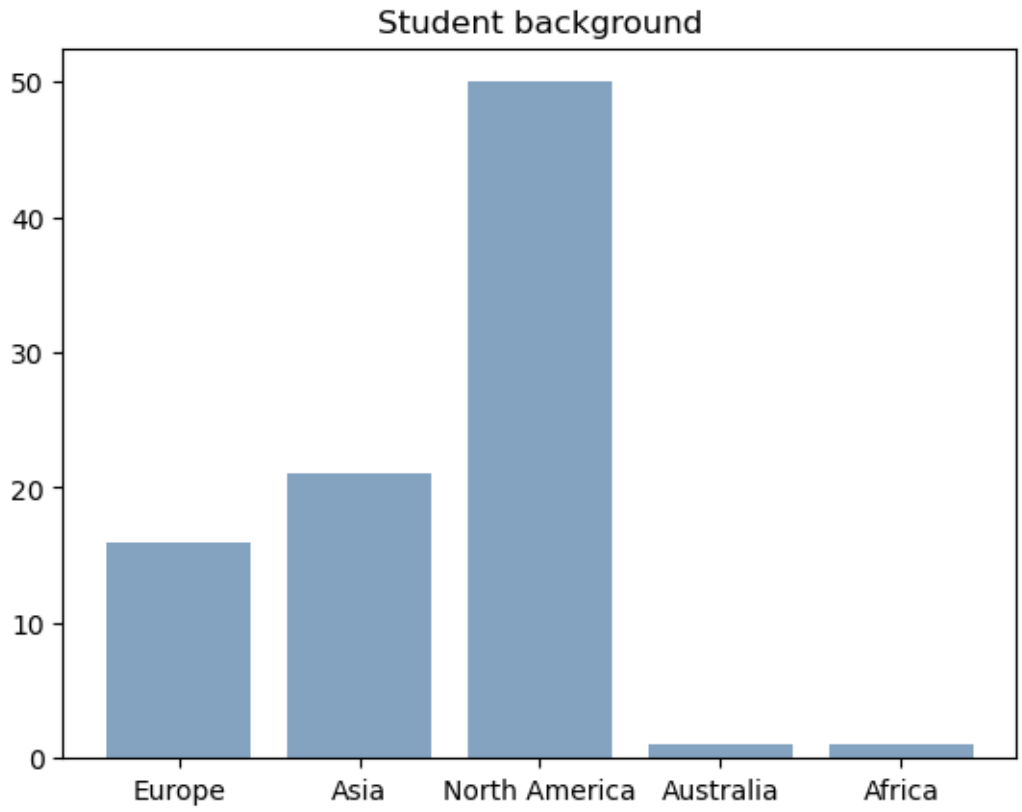
Given the data `y_pos` and `stu` defined for you in the code cell below, use the `matplotlib` function `bar()` to create a bar plot. Inside the `bar()` function, set the argument `color` equal to (0.2, 0.4, 0.6, 0.6). Make the following customizations to the plot:

- Set the title equal to "Student background"
- Set the argument of the function plt.xticks equal to (y_pos, bars).

Save your plot as a png file with the name "plot2.png" in the folder "results".

[6]:
```
### GRADED

# Data
bars = ('Europe', 'Asia', 'North America', 'Australia', 'Africa')
y_pos = np.arange(len(bars))
stu = [16, 21, 50, 1, 1]
```

```
### YOUR SOLUTION HERE


plt.bar(bars, stu, color=(0.2, 0.4, 0.6, 0.6))
plt.title("Student background")
plt.xticks(y_pos, bars)
plt.savefig("results/plot2.png")
plt.show()

###
### YOUR CODE HERE
###
```



[7]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.1.4 Question 3

In the code cell below, we have defined for you the necessary data to construct this plot.

Given the data `r1` and `bars1` create a first bar plot by making sure you set the following parameters: `width=bar_width, color='blue', edgecolor='black', yerr=yer1, capsize=7, label='Female'`.

Next, on the same graph, given the data below r2 and bars2 create a second bar plot by making sure you set the following parameters: `width=bar_width, color='cyan', edgecolor='black', yerr=yer2, capsize=7,label='Male'`.

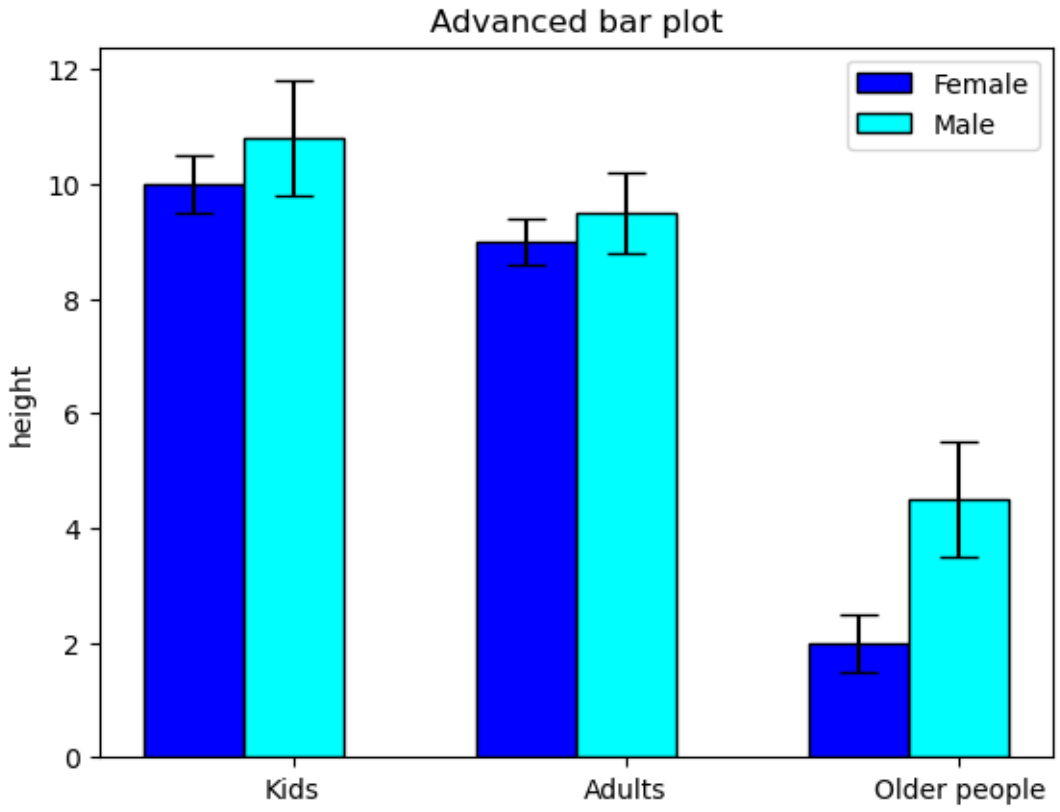
Finally, use the appropriate matplotlib functions to add the following elements to the plot:

- Title: "Advanced bar plot"
- Y-label = "height"
- Legend = Show a standard label with the name of the bars. **Hint: Use the function plt.legend() to show the legend.**

Save your plot as a png file with the name "plot3.png" in the folder "results".

[8]:
```
### GRADED

# Data

# width of the bars
bar_width = 0.3

# Choose the height of the blue bars
bars1 = [10, 9, 2]

# Choose the height of the cyan bars
bars2 = [10.8, 9.5, 4.5]

# Choose the height of the error bars (bars1)
yer1 = [0.5, 0.4, 0.5]

# Choose the height of the error bars (bars2)
yer2 = [1, 0.7, 1]

# The x position of bars
r1 = np.arange(len(bars1))
r2 = [x + bar_width for x in r1]

# general layout
plt.xticks([r + bar_width for r in range(len(bars1))], ['Kids', 'Adults',␣
 ↪'Older people'])

### YOUR SOLUTION HERE

plt.bar(r1, bars1,width=bar_width, color='blue', edgecolor='black', yerr=yer1,␣
 ↪capsize=7, label='Female')
```

```
plt.bar(r2, bars2,width=bar_width, color='cyan', edgecolor='black', yerr=yer2,␣
  ↪capsize=7,label='Male')
plt.title("Advanced bar plot")
plt.ylabel("height")
plt.legend()

plt.savefig("results/plot3.png")
plt.show()


###
### YOUR CODE HERE
###
```



[9]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.1.5 Question 4

In the code cell below, we have set the values of `N_points` and `n_bins` for you, as well as a random seed generator for reproducibility.

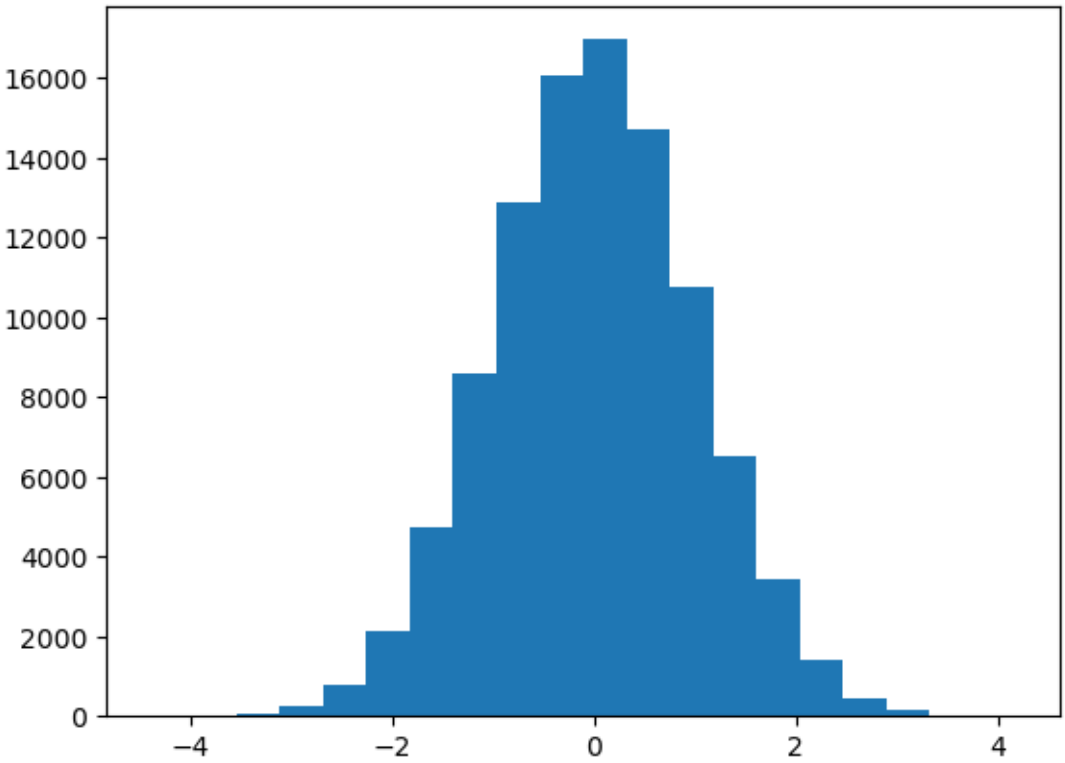
Generate the data `x` for a normal distribution by passing the argument `N_points` to the NumPy function `random.randn`.

Use the appropriate matplotlib function to generate a histogram for x. Set the parameter `bins = n_bins`.

Save your plot as a png file with the name "plot4.png" in the folder "results".

HINT: Generate x by using the command np.random.randn(N_points)

[10]:
```
### GRADED
import random
N_points = 100000
n_bins = 20
np.random.seed(123)

### YOUR SOLUTION HERE
x = np.random.randn(N_points)

plt.hist(x, bins = n_bins)

plt.savefig("results/plot4.png")
plt.show()
plt.close()

###
### YOUR CODE HERE
###
```



[11]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.2 Airbnb plots

1.2.1 Read the data

df = pd.read_csv("data/AB_NYC_2019.csv")

[12]:
```
df = pd.read_csv("/Users/dempseywade/Desktop/gitRepo/DartmouthCodingAssignments/
↪data/Mod7_AB_NYC_2019.csv")
```

[13]:
```
df.head()
```

[13]:
```
     id                                              name  host_id  \
0  2539                Clean & quiet apt home by the park     2787
1  2595                             Skylit Midtown Castle     2845
2  3647               THE VILLAGE OF HARLEM…NEW YORK !     4632
3  3831                   Cozy Entire Floor of Brownstone     4869
4  5022  Entire Apt: Spacious Studio/Loft by central park     7192

     host_name neighbourhood_group neighbourhood  latitude  longitude  \
```

```
0         John            Brooklyn    Kensington  40.64749  -73.97237
1     Jennifer           Manhattan       Midtown  40.75362  -73.98377
2    Elisabeth           Manhattan        Harlem  40.80902  -73.94190
3  LisaRoxanne            Brooklyn  Clinton Hill  40.68514  -73.95976
4        Laura           Manhattan   East Harlem  40.79851  -73.94399

         room_type  price  minimum_nights  number_of_reviews last_review  \
0     Private room    149               1                  9  2018-10-19
1  Entire home/apt    225               1                 45  2019-05-21
2     Private room    150               3                  0         NaN
3  Entire home/apt     89               1                270  2019-07-05
4  Entire home/apt     80              10                  9  2018-11-19

   reviews_per_month  calculated_host_listings_count  availability_365
0               0.21                               6               365
1               0.38                               2               355
2                NaN                               1               365
3               4.64                               1               194
4               0.10                               1                 0
```

[14]:
```
df.info()
```

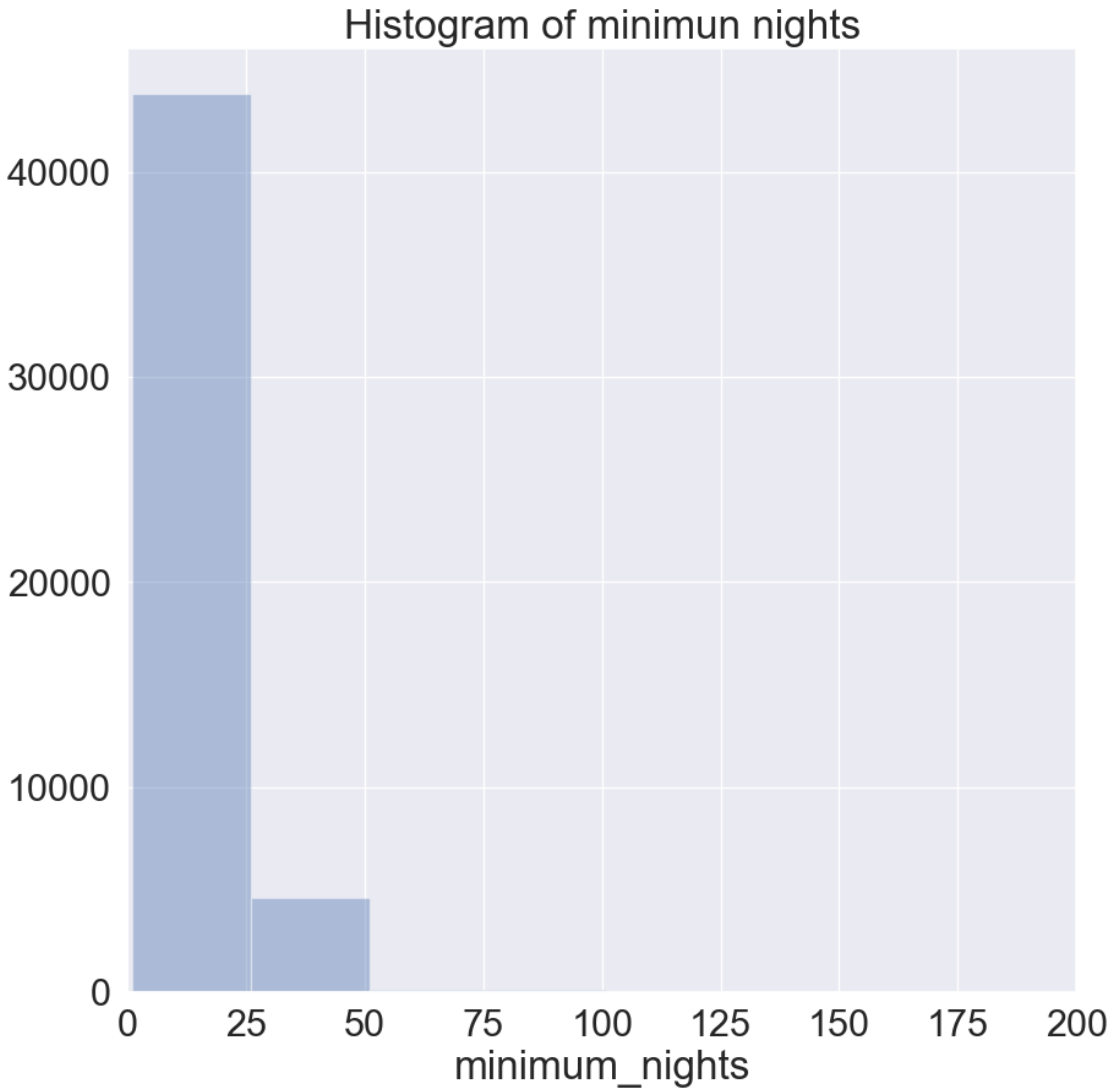
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              48895 non-null  int64
 1   name                            48879 non-null  object
 2   host_id                         48895 non-null  int64
 3   host_name                       48874 non-null  object
 4   neighbourhood_group             48895 non-null  object
 5   neighbourhood                   48895 non-null  object
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object
 9   price                           48895 non-null  int64
 10  minimum_nights                  48895 non-null  int64
 11  number_of_reviews               48895 non-null  int64
 12  last_review                     38843 non-null  object
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64
 15  availability_365                48895 non-null  int64
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```

1.2.2 Question 5

Use the function `figure()` to set the figure size to (10,10) and a `font_scale` of 2. Next, using `seaborn` function `distplot()`, create a histogram for the column `minimum_nights` by setting the argument `kde` equal to `False` so you only show the histogram. Set the limit for the x axis from 0 to 200. Finally add the title: "Histogram of minimun nights"

Save your plot as a png file with the name "plot5.png" in the folder "results".

[15]:
```
### GRADED

### YOUR SOLUTION HERE

sns.set(font_scale = 2)
plt.figure(figsize = (10,10))
sns.distplot(df['minimum_nights'], kde=False)
plt.xlim(0,200)
plt.title("Histogram of minimun nights")
plt.savefig("results/plot5.png")
#plt.close()

###
### YOUR CODE HERE
###
```



[16]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.2.3 Question 6

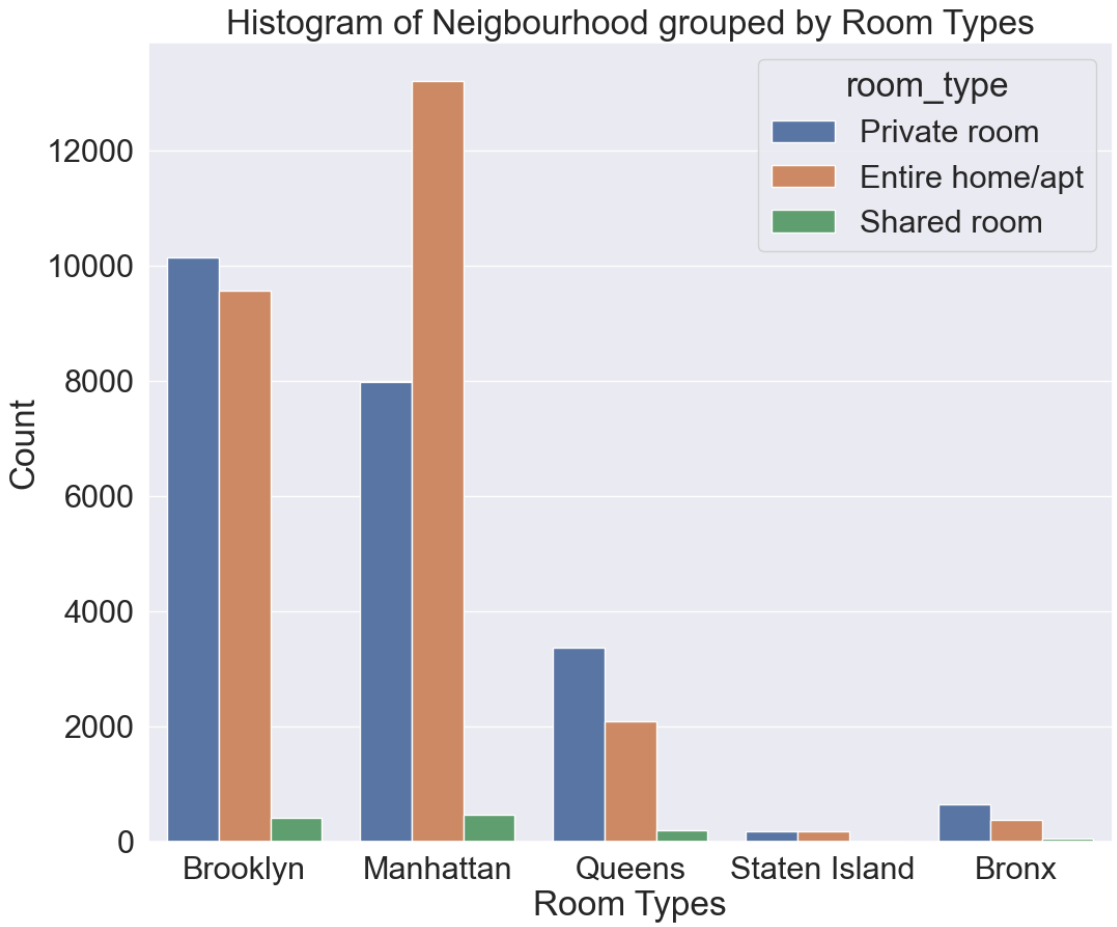
Use the `seaborn` function `countplot()` to create a histogram of the `neighbourhood_group` column and group it by `room_type`.

Use a a figsize of (12,10) and a font_scale of 2.

Add the title: "Histogram of Neigbourhood grouped by Room Types", ax x-label equal to "Room types" and a y-label equal to "Count".

Save your plot as a png file with the name "plot6.png" in the folder "results".

[17]:
```
### GRADED

plt.figure(figsize = (12,10))
sns.set(font_scale = 2)

#Use the seaborn function countplot() to create a histogram
#of the neighbourhood_group column and group it by
#room_type.
sns.countplot(x=df.neighbourhood_group, hue=df.room_type)

plt.title("Histogram of Neigbourhood grouped by Room Types")
plt.xlabel("Room Types")
plt.ylabel("Count")

plt.savefig("results/plot6.png")
plt.show()

### YOUR SOLUTION HERE
###
### YOUR CODE HERE
###
```



[18]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.2.4 Question 7

Using the `seaborn` function `barplot()`, create a sorted bar plot for the 5 top hosts (defined as people with the most listings). The plot shuld have the count for the host listings in the `y` axis and the name of the host in the `x` axis.

Use a a figsize of (12,10) and a font_scale of 2. Add the title: "Top 5 Hosts", set the `x-label` equal to "Host names" and the `y-label` equal to "Listings count".

Save your plot as a png file with the name "plot7.png" in the folder "results".

Hint: You have to drop duplicates from the columns 'host_name','calculated_host_listings_count' and then select the top 5.

[19]:
```
df.host_name.value_counts().head(5)
```

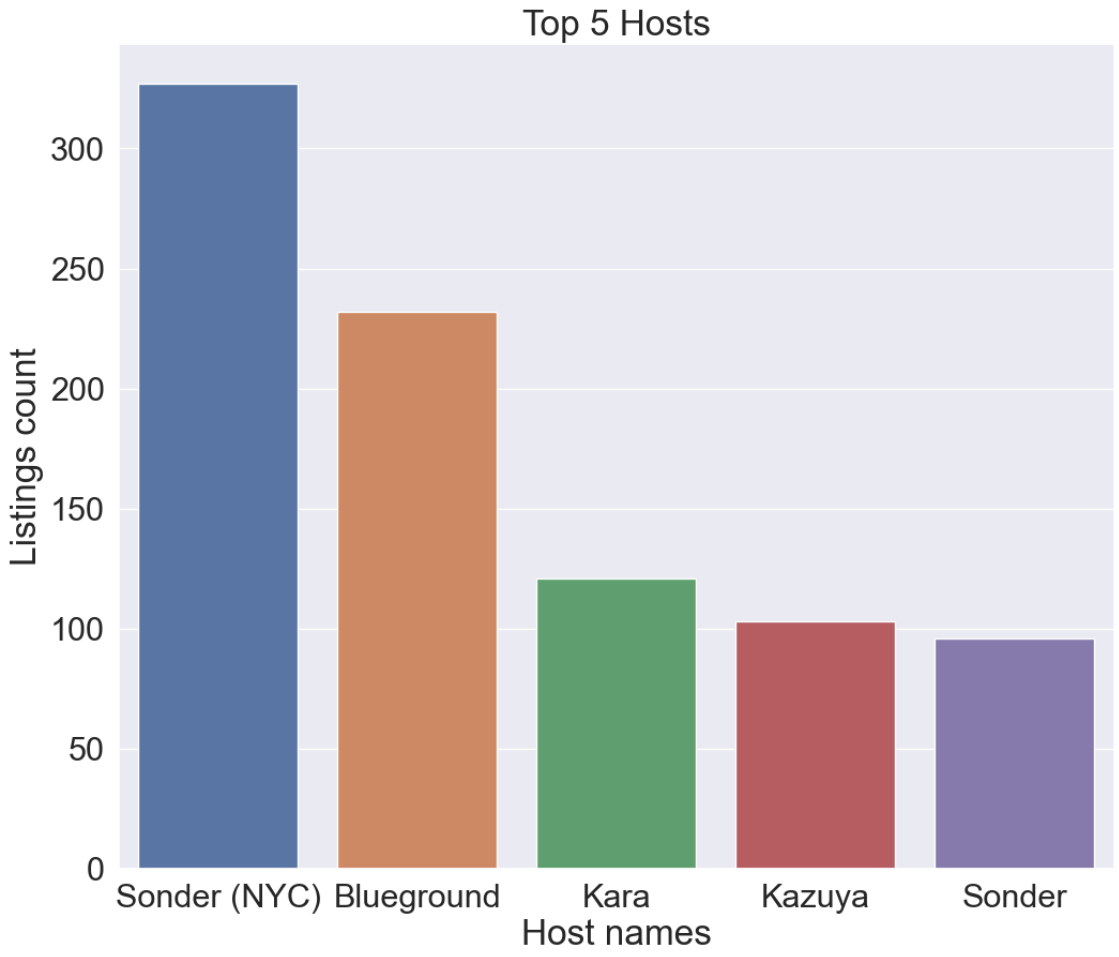
[19]:
```
Michael          417
David            403
Sonder (NYC)     327
John             294
Alex             279
Name: host_name, dtype: int64
```

[20]:
```
df_1 = df.sort_values(by=['calculated_host_listings_count'], ascending=False)
df_1 = df_1.drop_duplicates('calculated_host_listings_count')
df_final = df_1.head(5)

df_final.calculated_host_listings_count.head(5)
```

[20]:
```
39773    327
38701    232
13039    121
42840    103
33464     96
Name: calculated_host_listings_count, dtype: int64
```

[21]:
```
### GRADED

### YOUR SOLUTION HERE

plt.figure(figsize=(12,10))
sns.set(font_scale=2)

#sns.barplot(df7.host_name, y=df7.calculated_host_listings_count)
df_1 = df.sort_values(by=['calculated_host_listings_count'], ascending=False)
df_1 = df_1.drop_duplicates('calculated_host_listings_count')
df_final = df_1.head(5)

ax = sns.barplot(x=df_final.host_name, y=df_final.
 ↪calculated_host_listings_count)
ax.set_title('Top 5 Hosts')
ax.set_xlabel('Host names')
ax.set_ylabel('Listings count')
plt.savefig('results/plot7.png')
plt.show()

#sns.barplot(x=df_final.host_name, y=df_final.calculated_host_listings_count)
#plt.xlabel('Host names')
#plt.ylabel('Listings count')
#plt.title('Top 5 Hosts')
#plt.savefig('results/plot7.png')
#plt.show()
```

```
###
### YOUR CODE HERE
###
```



[22]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

1.2.5 Question 8

Create a bar plot for the count of rooms (`room type`) per `neighbourhood_group` in the dataset.

Use a a figsize of (12,10) and a font_scale of 2. Add the title: "Room counts of Neighbourhood groups", set the `xlabel` equal "Neighbourhood Group" and the `y-label` equal to "Room Count".

Save your plot as a png file with the name "plot8.png" in the folder "results".

Hint: Do a groupby by `neighbourhood_group` selecting `as_index=False` and then do a count per `room_type`.

[23]:
```
df8 = df.groupby(['neighbourhood_group'], as_index=False).count()
df8
```

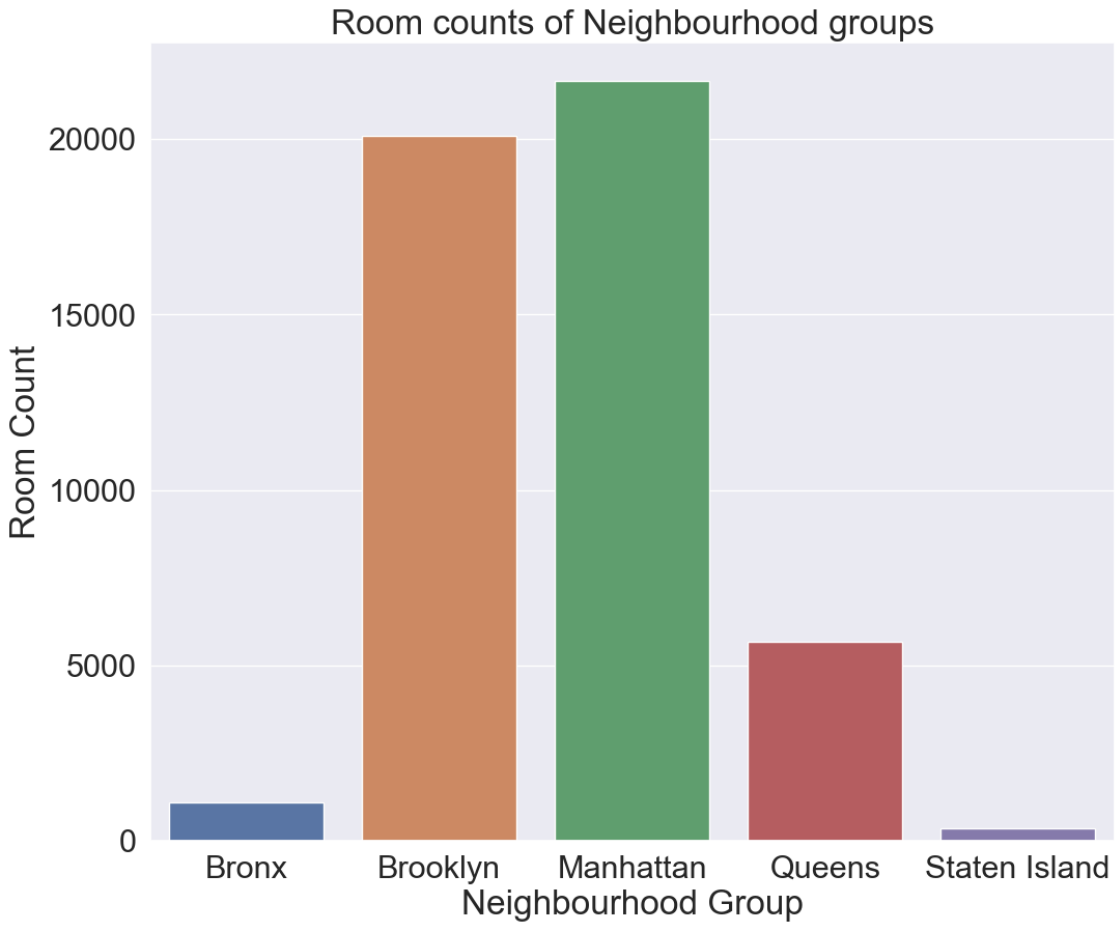
[23]:

	neighbourhood_group	id	name	host_id	host_name	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count	availability_365
0	Bronx	1091	1090	1091	1090	1091	1091	1091	1091	1091	1091	1091	876	876	1091	1091
1	Brooklyn	20104	20098	20104	20095	20104	20104	20104	20104	20104	20104	20104	16447	16447	20104	20104
2	Manhattan	21661	21652	21661	21652	21661	21661	21661	21661	21661	21661	21661	16632	16632	21661	21661
3	Queens	5666	5666	5666	5664	5666	5666	5666	5666	5666	5666	5666	4574	4574	5666	5666
4	Staten Island	373	373	373	373	373	373	373	373	373	373	373	314	314	373	373

[ ]:
```

```

[24]:
```
### GRADED

### YOUR SOLUTION HERE

plt.figure(figsize = (12,10))
sns.set(font_scale = 2)
#df8 = group by neighborhood_group as index=False select room_type and do a␣
 ↪count

df8 = df.groupby(['neighbourhood_group'], as_index=False).count()

ax = sns.barplot(x=df8.neighbourhood_group, y=df8.room_type)
ax.set_title('Room counts of Neighbourhood groups')
ax.set_xlabel('Neighbourhood Group')
```

```
ax.set_ylabel('Room Count')
plt.savefig('results/plot8.png')
plt.show()
#plt.close

#barplot

###
### YOUR CODE HERE
###
```



[25]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```

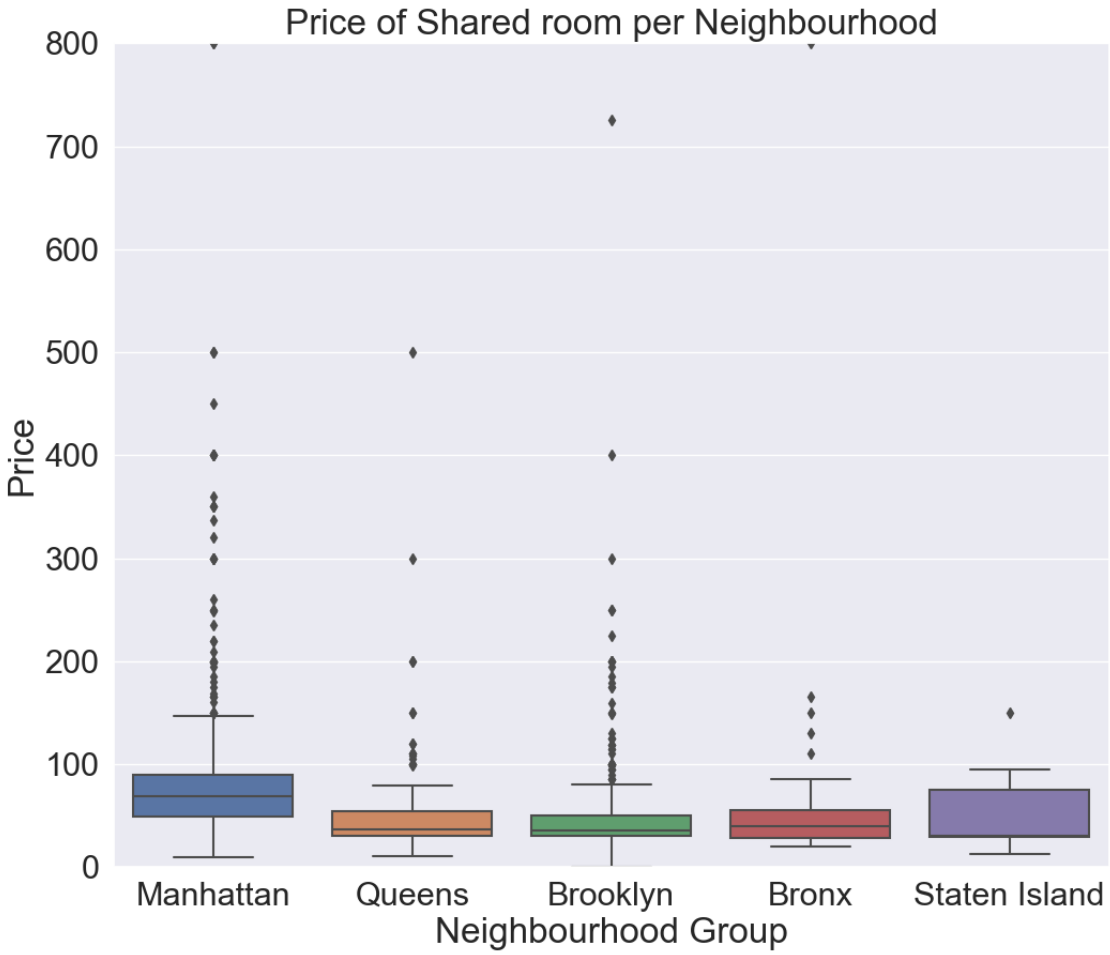
1.2.6 Question 9

Use the `seaborn` function `boxplot()` to create a box plot for the price of a `Shared Room` per `neighbourhood_group` in the dataset.

Use a a figsize of (12,10) and a font_scale of 2. Add the title: "Price of Shared room per Neighbourhood". Set the `x-label` equal to "Neighbourhood Group" and the `y-label`equal "Price". Finally set a `ylim` from 0 to 800.

Save your plot as a png file with the name "plot9.png" in the folder "results".

[26]:
```
### GRADED

### YOUR SOLUTION HERE
plt.figure(figsize=(12,10))
sns.set(font_scale = 2)

#df9 = selecting where room_type is equal to shared room
df9 = df.loc[df['room_type'] =='Shared room']

ax = sns.boxplot(x=df9['neighbourhood_group'], y=df9['price'])

ax.set_ybound(0,800)

ax.set_title('Price of Shared room per Neighbourhood')

ax.set_xlabel('Neighbourhood Group')

ax.set_ylabel('Price')

plt.savefig('results/plot9.png')
plt.show()
#plt.close()
###
### YOUR CODE HERE
###
```



[27]:
```
###
### AUTOGRADER TEST - DO NOT REMOVE
###
```