**Group Details:**


- Name: ID: 322262361 אדר חזאזי גרש
- Name: ID: 208252882 ליעם קלס
- Name: ID: 323716365 עינב טולדו


## Helper Functions and Imports

```
#@title Helper Functions and Imports

from pydrive2.auth import GoogleAuth
from google.colab import drive
from pydrive2.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib as mpl
import numpy as np
from scipy.stats import pearsonr, spearmanr

# Some visual settings
sns.set()
mpl.rcParams['xtick.labelsize'] = 12
mpl.rcParams['ytick.labelsize'] = 12
mpl.rcParams['axes.labelsize'] = 14

RENT_ID = '1R6v2uHpFyNb1z2DT0M_JHTUE3PHFFYmu'
SOCIORANK_ID = '1gc57mT5zgIb-XeVsMfCphnWTRz1-dmLj'

def load_df(drive_id, **load_kwargs):
  auth.authenticate_user()
  gauth = GoogleAuth()
  gauth.credentials = GoogleCredentials.get_application_default()
  drive = GoogleDrive(gauth)
  download = drive.CreateFile({'id': drive_id})
  filename = '{}.csv'.format(drive_id)
  download.GetContentFile(filename)
  return pd.read_csv(filename, **load_kwargs)
```

# Introduction to Data Science - Lab #2

## Exploratory Data Analysis

## Case Study: Rental Listings in Jerusalem

In this lab we will practice our exploratory data analysis skills using real data!

We will explore data of rental pricings in Jersualem. The dataset consists of listings published in https://www.komo.co.il/ during the summer of 2022.

We will use two python packages for visualizing the data: `matplotlib` (and specifically its submodule `pyplot` imported here as `plt`) and `seaborn` (imported as `sns`). Seaborn is a package that "wraps" matplotlib and introduces more convenient functions for quickly creating standard visualizations based on dataframes.

Please **breifly** go over this quick start guide to matplotlib, the first seaborn introduction page until the "Multivariate views on complex datasets" section (not included), and the second introduction page until the "Combining multiple views on the data" section.

## Loading the dataset

```
#@title Loading the dataset
rent_df = load_df(RENT_ID)[['propertyID','neighborhood','monthlyRate','mefarsem','room
rent_df = rent_df.drop_duplicates(subset='propertyID').reset_index(drop=True)
rent_df_backup_for_exercise = rent_df.copy()
clean_df_area_filtered = None
clean_df = None
```

Let's print a random sample:

```
np.random.seed(2)
rent_df.sample(5)
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | entr |
|---|---|---|---|---|---|---|---|---|
| **403** | 3981729 | גבעת מרדכי | 4500.0 | private | 3.0 | 6.0 | 62.0 | 10/08/202 |
| **457** | 3991612 | קריית משה | 3000.0 | private | 3.5 | 3.0 | NaN | Nal |

And print some summary statistics:

```
rent_df.describe(include='all')
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| **count** | 6.120000e+02 | 612 | 612.000000 | 612 | 612.000000 | 611.000000 |
| **unique** | NaN | 54 | NaN | 2 | NaN | NaN |
| **top** | NaN | קריית יובל | NaN | private | NaN | NaN |
| **freq** | NaN | 66 | NaN | 600 | NaN | NaN |
| **mean** | 3.981582e+06 | NaN | 4717.393791 | NaN | 2.927288 | 1.916530 |
| **std** | 6.525543e+04 | NaN | 2195.215139 | NaN | 1.007350 | 1.581006 |
| **min** | 2.494041e+06 | NaN | 0.000000 | NaN | 1.000000 | -2.000000 |
| **25%** | 3.981694e+06 | NaN | 3500.000000 | NaN | 2.000000 | 1.000000 |
| **50%** | 3.987901e+06 | NaN | 4400.000000 | NaN | 3.000000 | 2.000000 |
| **75%** | 3.992605e+06 | NaN | 5800.000000 | NaN | 3.500000 | 3.000000 |

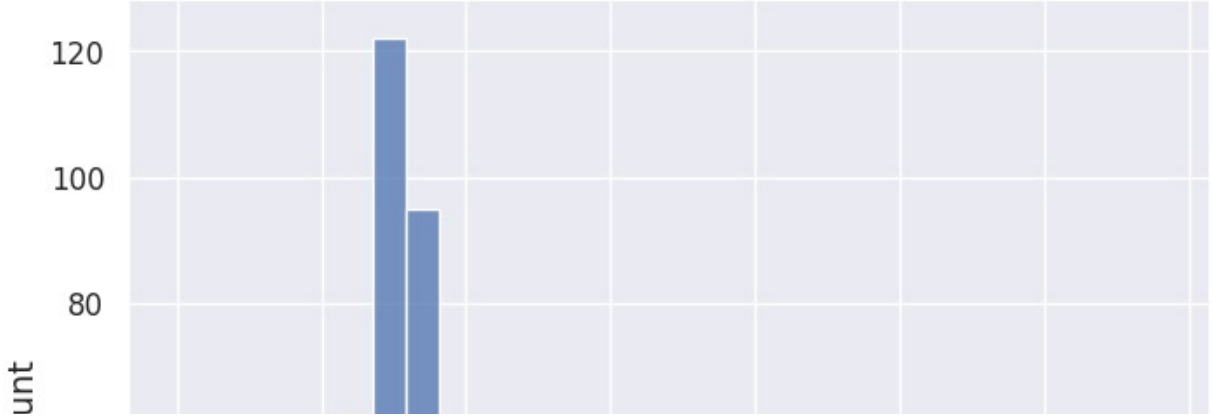
The variables we will focus on are:

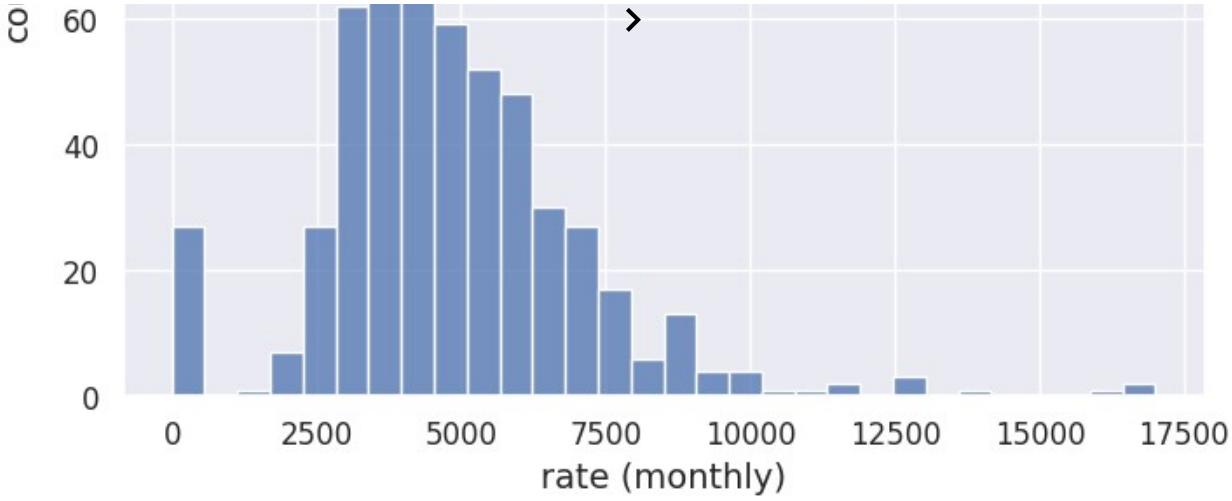
1. neighborhood: The hebrew name of the neighborhood in jerusalem where the listing is located
2. monthlyRate: The monthly rate (שכר דירה) in shekels
3. rooms: The number of rooms in the apartment
4. floor: The floor in which the apartment is located
5. area: The area of the apartment in squared meters
6. numFloors: The total number of floors in the building

**What is the distribution of prices in this dataset?**

Q: Plot a histogram with 30 bins of the monthly rates in this dataset:

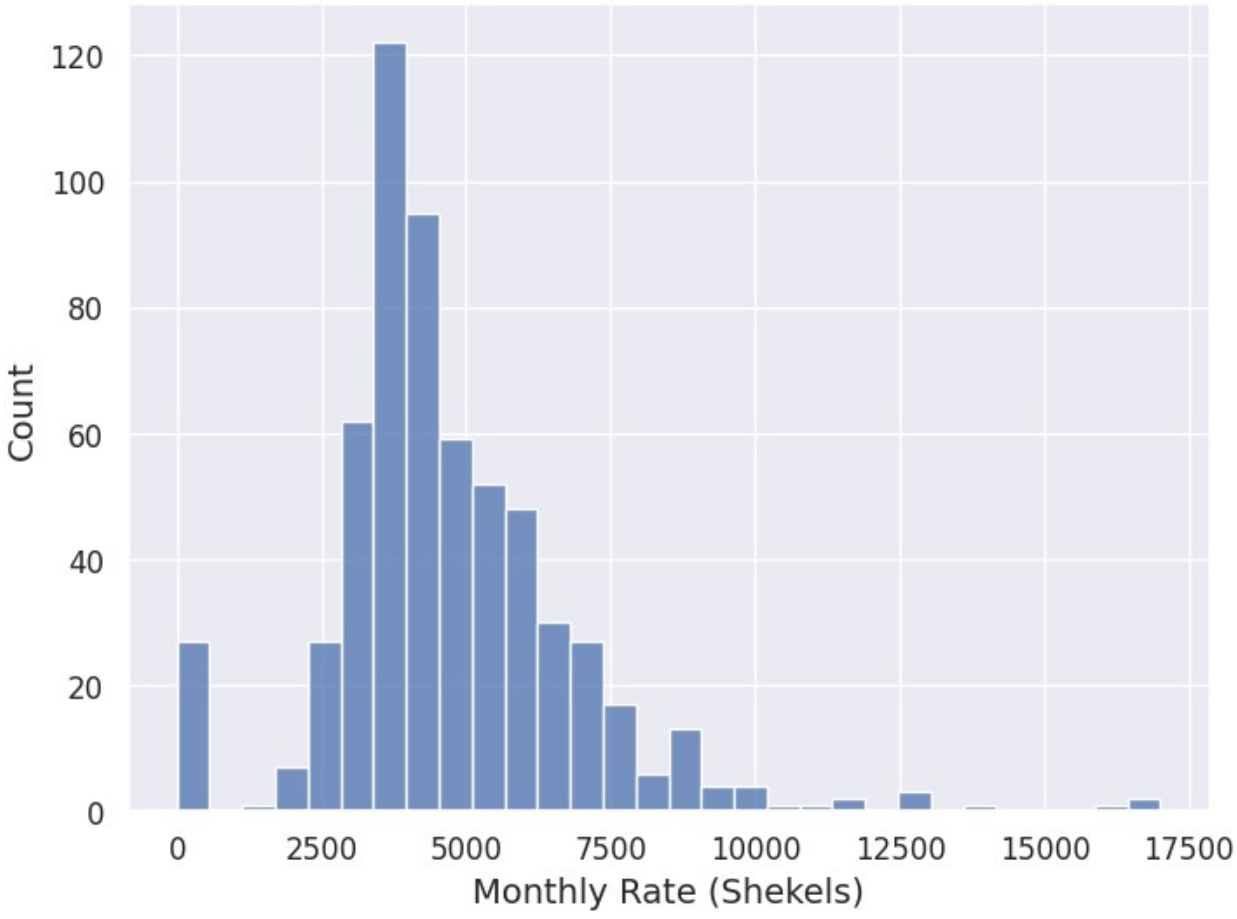
```
fig = plt.figure(figsize=(8,6))
sns.histplot(x='monthlyRate', data=rent_df, bins=30)
plt.xlabel("rate (monthly)")
plt.ylabel("count");
```





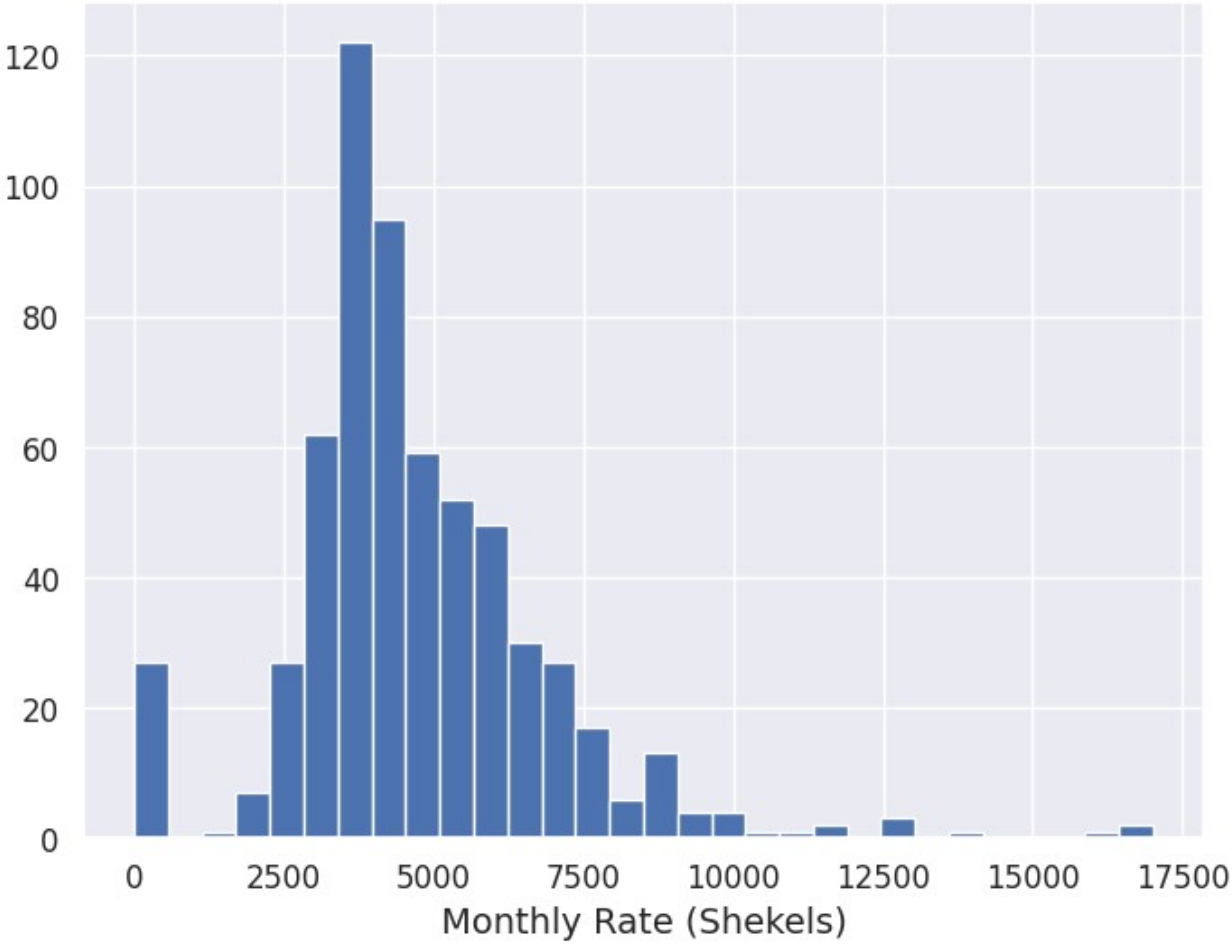

## Solution 1

Show code



## Solution 2

```
# @title Solution 2
rent_df["monthlyRate"].hist(bins=30, figsize=(8,6))
plt.xlabel("Monthly Rate (Shekels)");
```




We see that the prices distribution peaks around ~3500 Shekels and that it is right skewed, as there are some very expensive apartments. We can also see a peak at zero which makes sense as sometimes listings do not include a price. We would want to filter those out when we analyze prices later on.

Q: Print the number of listings that have no monthly rate:

## Solution

```
# @title Solution
print("Number of apartments without a price: ", rent_df['monthlyRate'].value_counts()[
```

```
Number of apartments without a price:  25
```

We want to remove those listings, but we don't want to lose these entries, as we might want to know how many and what type of outliers we originally removed. So we create another dataframe that has the listings we removed and the reason for removal.

```
outlier_df = pd.DataFrame(columns=rent_df.columns.to_list()+['reason']) # will save th

outliers = rent_df[rent_df['monthlyRate'] <= 0].reset_index(drop=True)
outliers['reason']= "monthlyRate <= 0"
outlier_df = pd.concat([outlier_df, outliers], axis=0, ignore_index=True).drop_duplica
outlier_df.tail()
```

Show hidden output

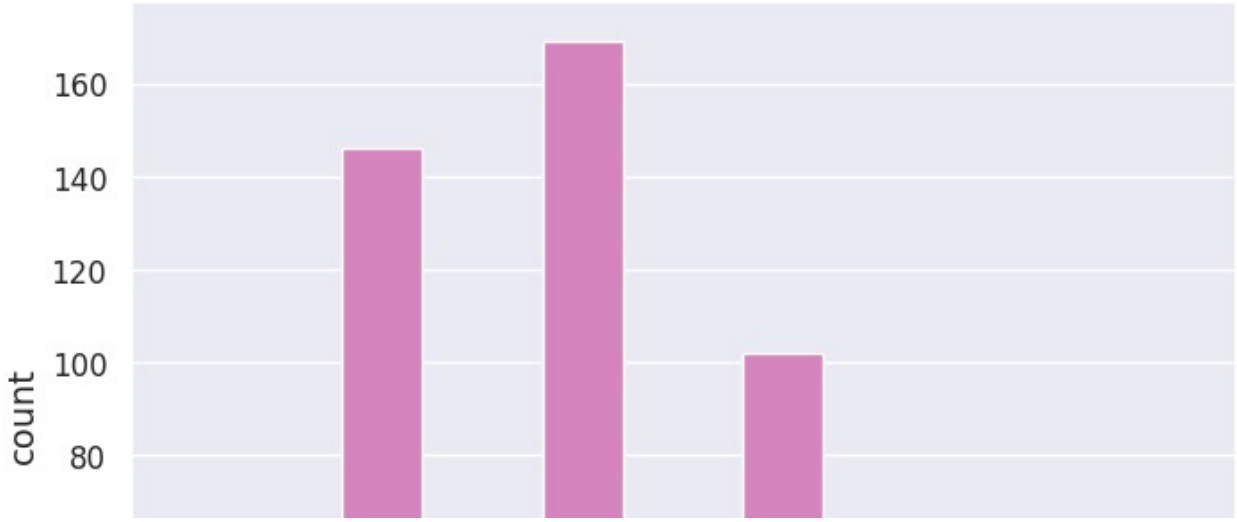
We will now remove those listings and save the result to a new variable `clean_df`:

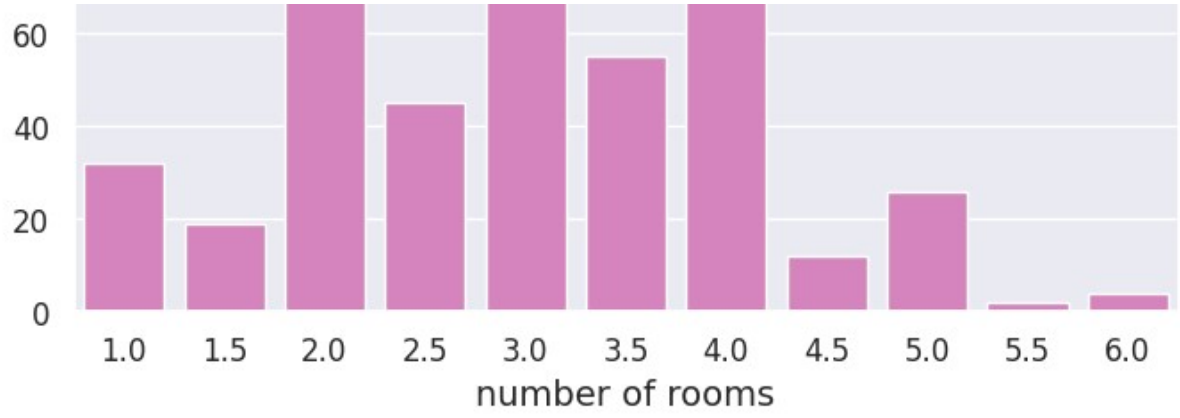
```
clean_df = rent_df[rent_df['monthlyRate'] > 0].reset_index(drop=True)
```

**What is the distribution of the number of rooms?**

Q: Use `sns.countplot` to compare the counts of listings with different numbers of rooms. Plot all bars in the same color of your choice.

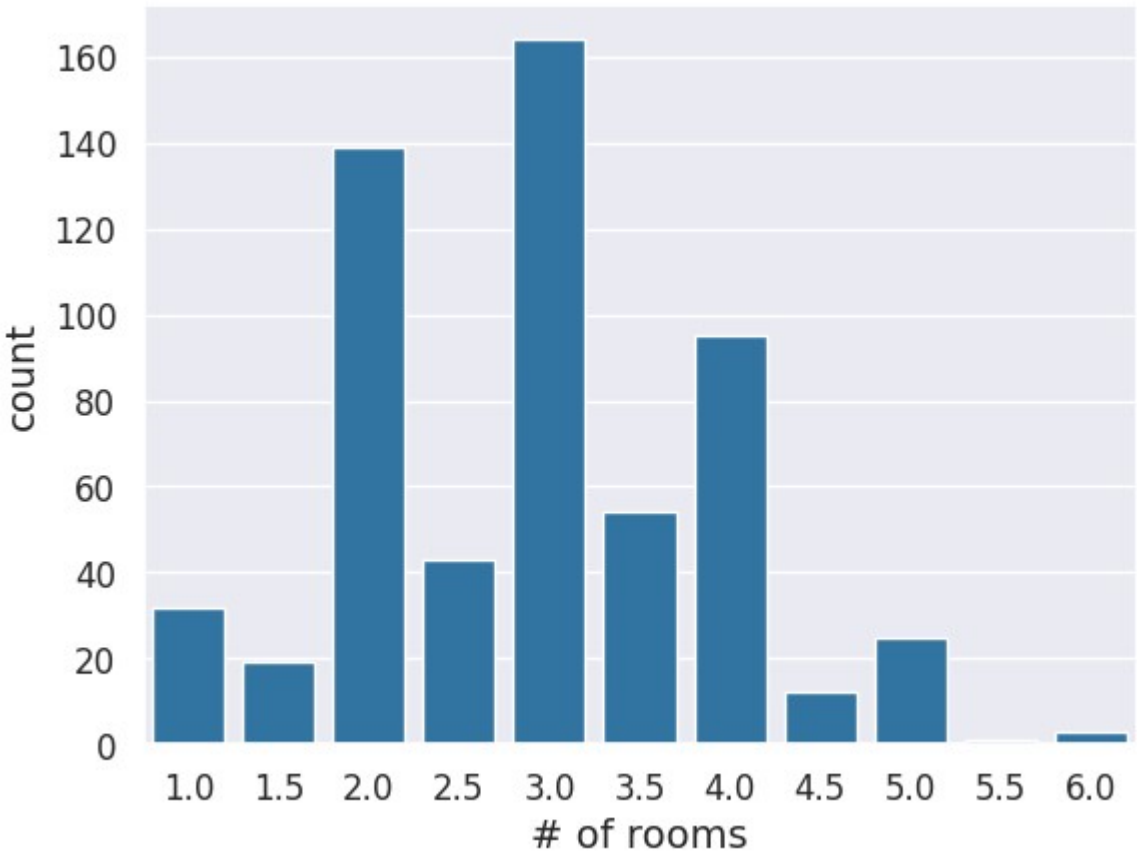
```
fig2 = plt.figure(figsize=(8,6))
sns.countplot(x='rooms', data=rent_df, color = 'tab:pink')
plt.xlabel("number of rooms")
plt.ylabel("count");
```






## Solution

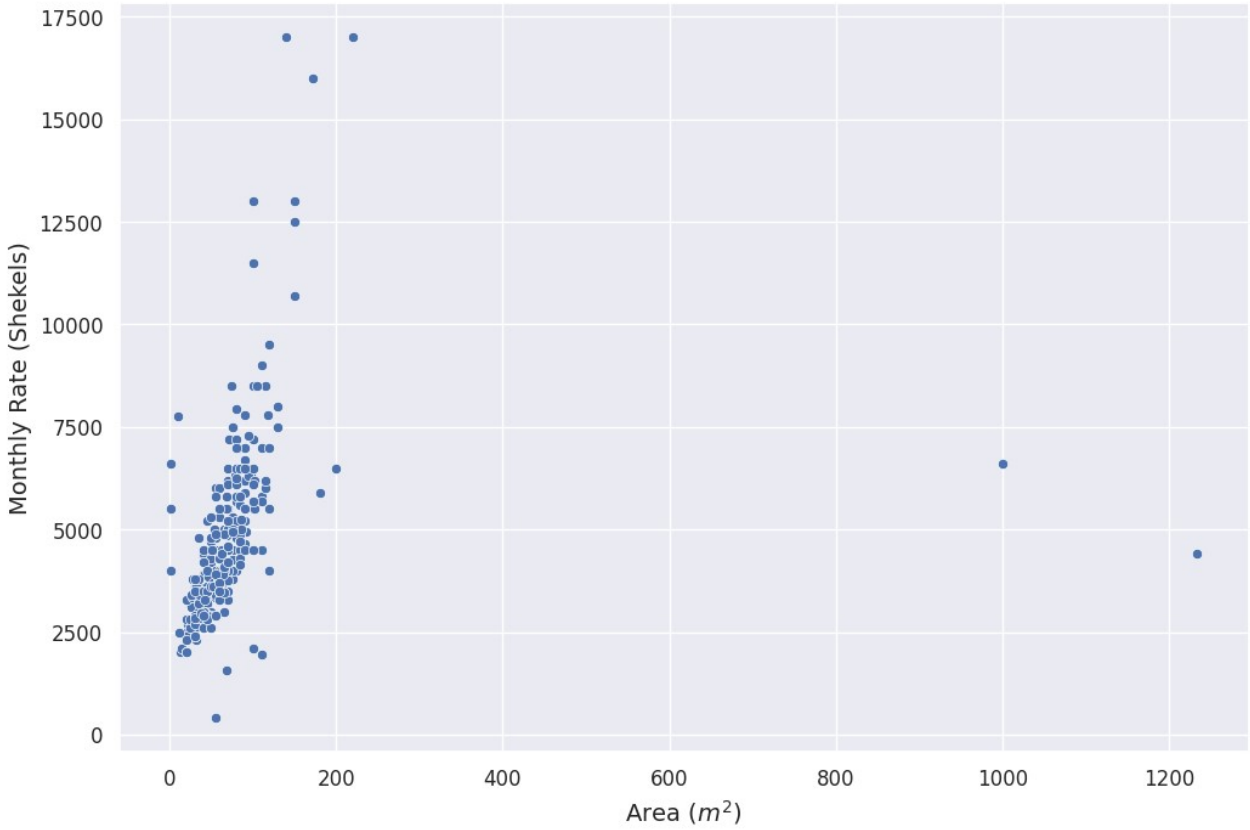
```
# @title Solution
if clean_df is None:
  print("Can't run until 'clean_df' is created!")
else:
  sns.countplot(x='rooms', data=clean_df, color='tab:blue')
  plt.xlabel("# of rooms");
```




The distribution peaks at three rooms and we also see that "half rooms" are less common.

**Can we see an association between apartment area and price?**

```
if clean_df is None:
  print("Can't run until 'clean_df' is created!")
else:
  plt.figure(figsize=(12,8))
  sns.scatterplot(x='area', y='monthlyRate', data=clean_df)
  plt.ylabel("Monthly Rate (Shekels)")
  plt.xlabel("Area ($m^2$)");
```




We see clear outliers here! We know that area is measured in squared meters and it is unlikely that there are any apartments of ~1000$m^2$.

Let's look at those samples to see if we can understand what happend there:

```
if clean_df is None:
```

```
if clean_df is None:
  print("Can't run until 'clean_df' is created!")
else:
  display(clean_df.sort_values('area', ascending=False).head(4))
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | ent |
|---|---|---|---|---|---|---|---|---|
| **185** | 3964340 | תלפיות | 4400.0 | private | 2.0 | 2.0 | 1234.0 | 10/08/20 |
| **543** | 3956561 | זכרון משה | 6600.0 | private | 3.5 | 3.0 | 1000.0 | 01/07/20 |

And inspect the description of one of those listings:

```
if clean_df is None:
  print("Can't run until 'clean_df' is created!")
else:
  display(clean_df.at[543,'description'])
```

```
'דירה מהממת בלב ירושלים. צמודה לרכבת הקלה- תחנת הדוידקה. 3 חדרים ענקיים ולכל
'חדר מרפסת גדולה. חלל כניסה עם פינת עבודה. מתאימה מאוד ל- 3 שותפות
```

Clearly not a 1000 m^2 apartment...

Double-click (or enter) to edit

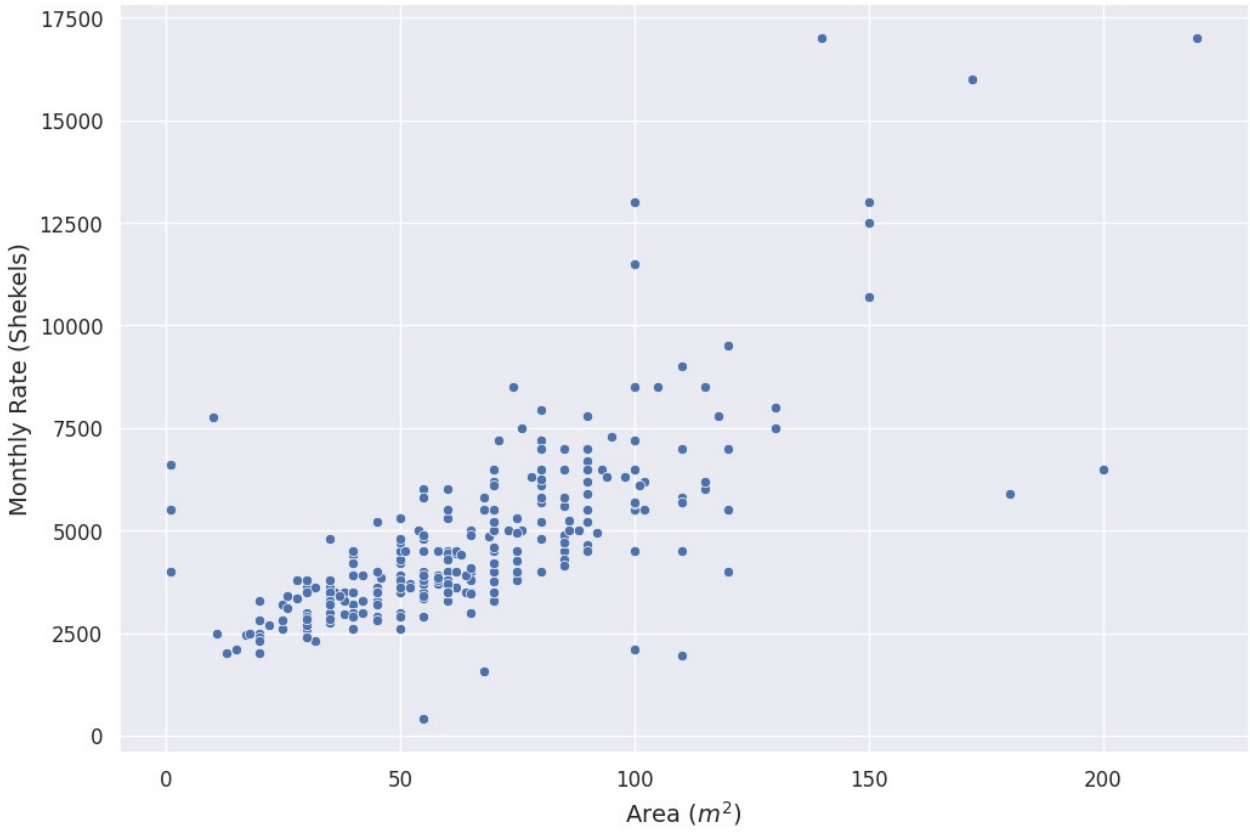
Q: Save a new dataframe named `clean_df_area_filtered` with all listings with area smaller than 800 m^2. Again, add the removed outliers to the outliers_df dataframe.

Plot again the scatter of area vs. monthly rate after removing the outliers.

```
clean_df_area_filtered = clean_df[clean_df["area"] < 800]
plt.figure(figsize=(12,8))
sns.scatterplot(x='area', y='monthlyRate', data=clean_df_area_filtered)
plt.ylabel("Monthly Rate (Shekels)")
plt.xlabel("Area ($m^2$)");

outliers = clean_df[clean_df['area'] > 800].reset_index(drop=True)
outliers['reason']= "area > 800"
outlier_df = pd.concat([outlier_df, outliers], axis=0, ignore_index=True).drop_duplica
outlier_df.tail()
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | entr |
|---|---|---|---|---|---|---|---|---|
| **22** | 3952750 | מוסררה | 0.0 | private | 5.5 | 1.0 | 180.0 | 10/08/202 |

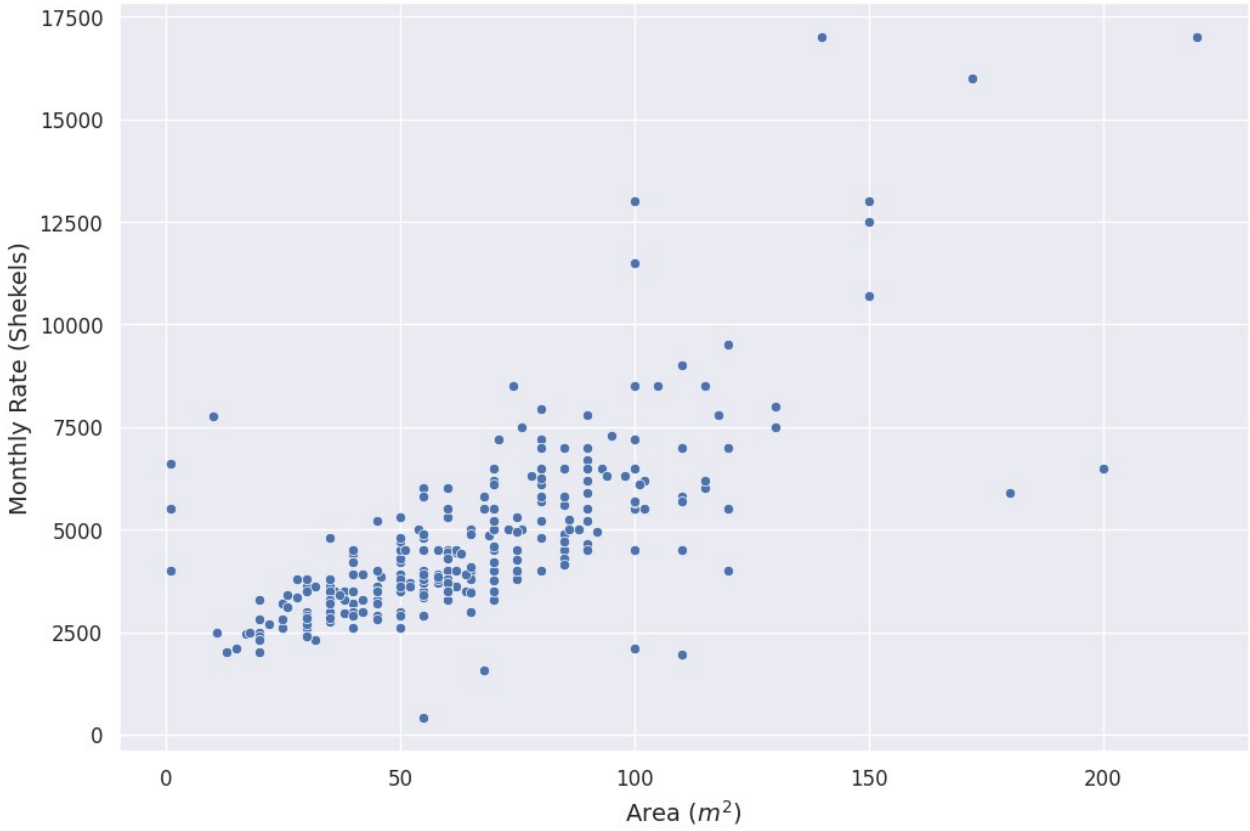
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **23** | 3988157 | גבעת שאול | 0.0 | private | 5.0 | 1.0 | 140.0 | 10/08/202 |
| **24** | 3981160 | גבעת משואה | 0.0 | private | 6.0 | -2.0 | NaN | Na |
| **25** | 3964340 | תלפיות | 4400.0 | private | 2.0 | 2.0 | 1234.0 | 10/08/202 |
| **26** | 3956561 | זכרון משה | 6600.0 | private | 3.5 | 3.0 | 1000.0 | 01/07/202 |




## Solution

```
# @title Solution
if clean_df is None:
  print("Can't run until 'clean_df' is created!")
elif outlier_df is None:
  print("Can't run until 'outlier_df' is created!")
else:
  # save outliers
  outliers = clean_df[clean_df['area'] >= 800].reset_index(drop=True)
  outliers['reason']= "'area' >= 800"
  outlier_df = pd.concat([outlier_df, outliers], axis=0, ignore_index=True).drop_dupli
```

```
  # remove the outliers from the dataset
clean_df_area_filtered = clean_df[clean_df['area'] < 800].reset_index(drop=True)
plt.figure(figsize=(12,8))
sns.scatterplot(x='area', y='monthlyRate', data=clean_df_area_filtered)
plt.xlabel("Area ($m^2$)")
plt.ylabel("Monthly Rate (Shekels)");
```




Again, we see some strange behavior of apartments with almost zero area but with a high monthly rate. Let's check them out:

We start with all apartments with an area between 0 to 25 $m^2$:

```
# Show all apartments with area between 0 and 25
```

```
clean_df_area_filtered[clean_df_area_filtered['area'].between(0,25)]
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | entry |
|---|---|---|---|---|---|---|---|---|
| **0** | 3994505 | קריית יובל | 2000.0 | private | 1.0 | 2.0 | 13.0 | 10/08/202 |
| **1** | 3981298 | רחביה | 2450.0 | private | 1.0 | 1.0 | 17.0 | 10/08/202 |
| **3** | 3993997 | בית וגן | 2100.0 | private | 1.0 | 0.0 | 15.0 | 10/08/202 |
| **5** | 3993552 | הר נוף | 2000.0 | private | 1.0 | 0.0 | 20.0 | 10/08/202 |
| **6** | 3972039 | גבעת שאול | 2700.0 | private | 1.0 | 0.0 | 22.0 | 10/08/202 |
| **7** | 3988096 | המושבה הגרמנית | 2500.0 | private | 1.0 | 0.0 | 18.0 | 10/08/202 |
| **8** | 3992809 | נחלאות | 3200.0 | private | 1.0 | 2.0 | 25.0 | 10/08/202 |
| **10** | 3983516 | הגבעה הצרפתית | 2000.0 | private | 1.0 | 2.0 | 20.0 | 10/08/202 |

Some make sense and others do not. Let's focus on the expensive ones (between 5,000 and 10,000 shekels):

```
# Show all apartments with area between 0 and 25 that also have a price between 5000 a
if clean_df_area_filtered is None:
  print("Can't run until 'clean_df_area_filtered' is created!")
else:
  display(clean_df_area_filtered[clean_df_area_filtered['area'].between(0,25) & clean_
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | entry |
|---|---|---|---|---|---|---|---|---|

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **197** | 3984483 | ארנונה | 6600.0 | private | 4.0 | 2.0 | 1.0 | 01/09/202 |

Those are clearly wrong too... Besides that the relationship between the area and the price seems linear. Let's remove these outliers too:

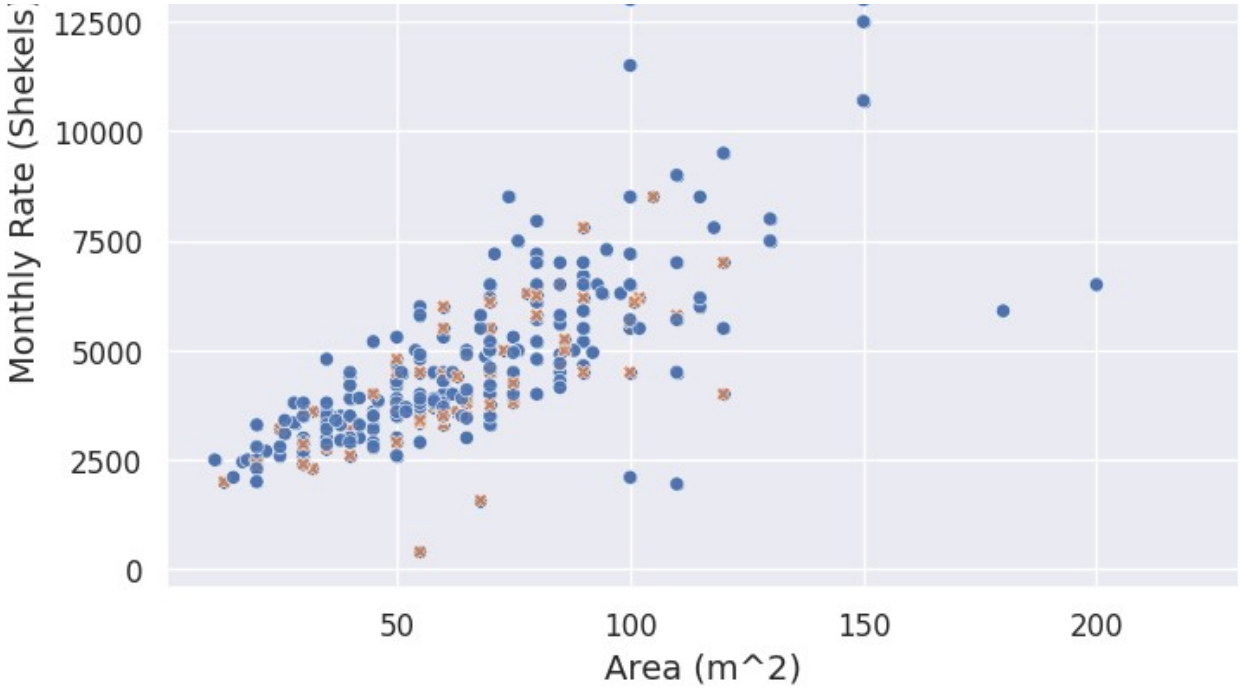
```
#remove the outliers
if clean_df_area_filtered is None:
  print("Can't run until 'clean_df_area_filtered' is created!")
elif outlier_df is None:
  print("Can't run until 'outlier_df' is created!")
else:
  non_ouliers = clean_df_area_filtered['area'] > 10 # get non outliers series of true/

  # save outliers
  outliers = clean_df_area_filtered[~non_ouliers].reset_index(drop=True) # get the out
  outliers['reason']= "'area' <= 10"
  outlier_df = pd.concat([outlier_df, outliers], axis=0, ignore_index=True).drop_dupli

  # remove them
  clean_df_area_filtered = clean_df_area_filtered[non_ouliers].reset_index(drop=True)
```

## Can we see a different pattern for top floor apartments?

Q: Plot again a scatter of area vs. monthly rate. This time distinguish (by color / marker style or both) between apartments that are in the top floor and the rest of the apartments. (To do that you should create a new column in `clean_df_area_filtered` called `is top floor` and set it to 1 if the apartment is in the top floor and 0 otherwise.)

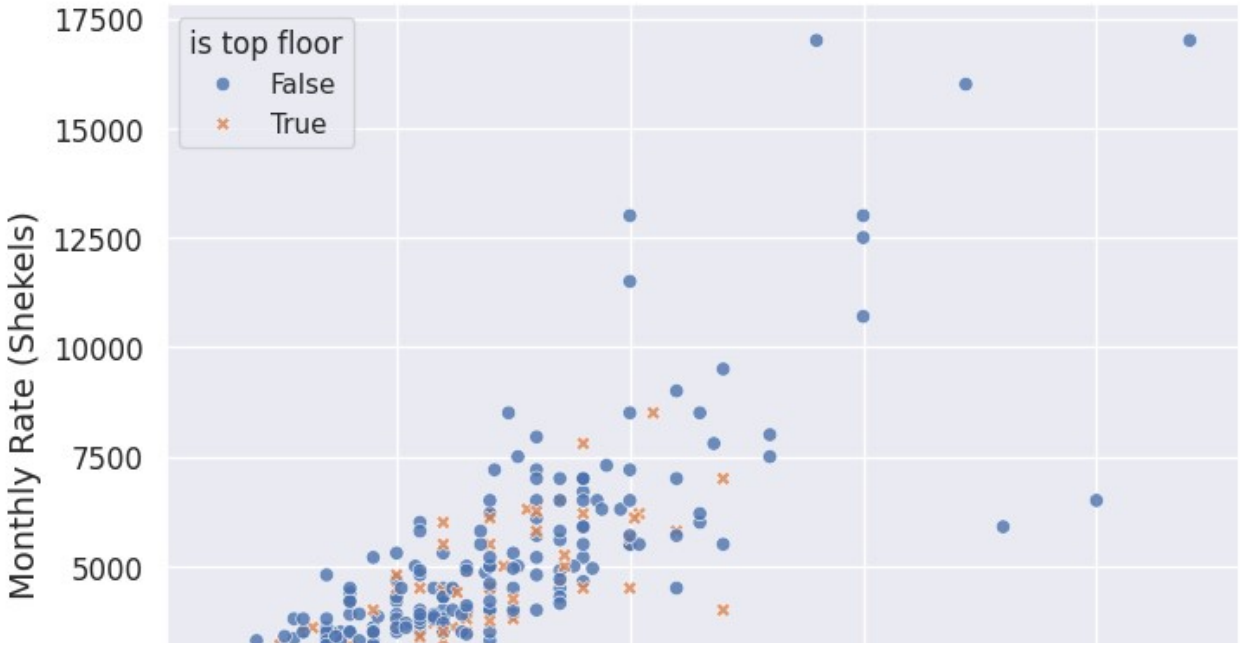
```
plt.figure(figsize = (8,6))
sns.scatterplot(x = "area", y = "monthlyRate", data = clean_df_area_filtered)
plt.xlabel("Area (m^2)")
plt.ylabel("Monthly Rate (Shekels)")

clean_df_area_filtered["is top floor"] = 0
clean_df_area_filtered.loc[clean_df_area_filtered['floor'] == clean_df_area_filtered['
#print(clean_df_area_filtered)
sns.scatterplot(x='area', y='monthlyRate', data=clean_df_area_filtered, alpha=0.8, hue
```

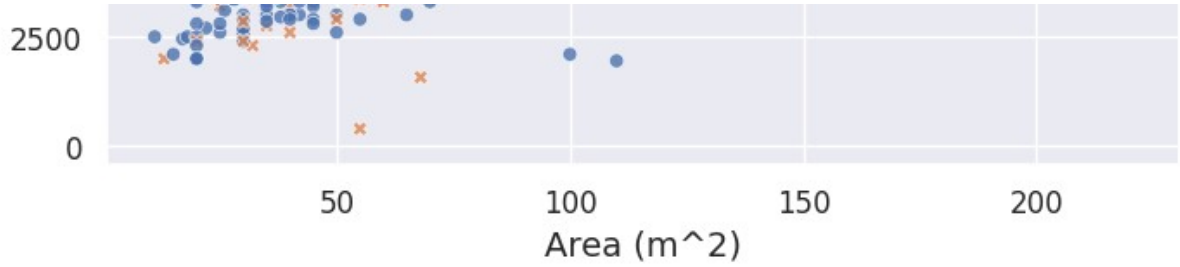
```
<Axes: xlabel='Area (m^2)', ylabel='Monthly Rate (Shekels)'>
```




## Solution

```
# @title Solution

if clean_df_area_filtered is None:
  print("Can't run until 'clean_df_area_filtered' is created!")
else:
  clean_df_area_filtered['is top floor'] = clean_df_area_filtered['floor'] == clean_df
  plt.figure(figsize=(8,6))
  sns.scatterplot(x='area', y='monthlyRate', data=clean_df_area_filtered, alpha=0.8, h
  plt.xlabel("Area (m^2)")
  plt.ylabel("Monthly Rate (Shekels)");
```






We can take a deeper look on the apartments with the very high monthly rate (to see if those are outliers or not):

```
if clean_df_area_filtered is None:
  print("Can't run until 'clean_df_area_filtered' is created!")
else:
  display(clean_df_area_filtered[clean_df_area_filtered['monthlyRate'] > 11000])
```

| | propertyID | neighborhood | monthlyRate | mefarsem | rooms | floor | area | entr |
|---|---|---|---|---|---|---|---|---|
| **198** | 3956418 | רחביה | 13000.0 | agent | 4.0 | 1.0 | 100.0 | Na |
| **236** | 3985051 | טלביה | 17000.0 | private | 4.0 | 4.0 | 140.0 | 10/08/202 |

We can see some representation of the more expensive neighborhoods of Jerusalem here.. More on the neighborhoods later on!

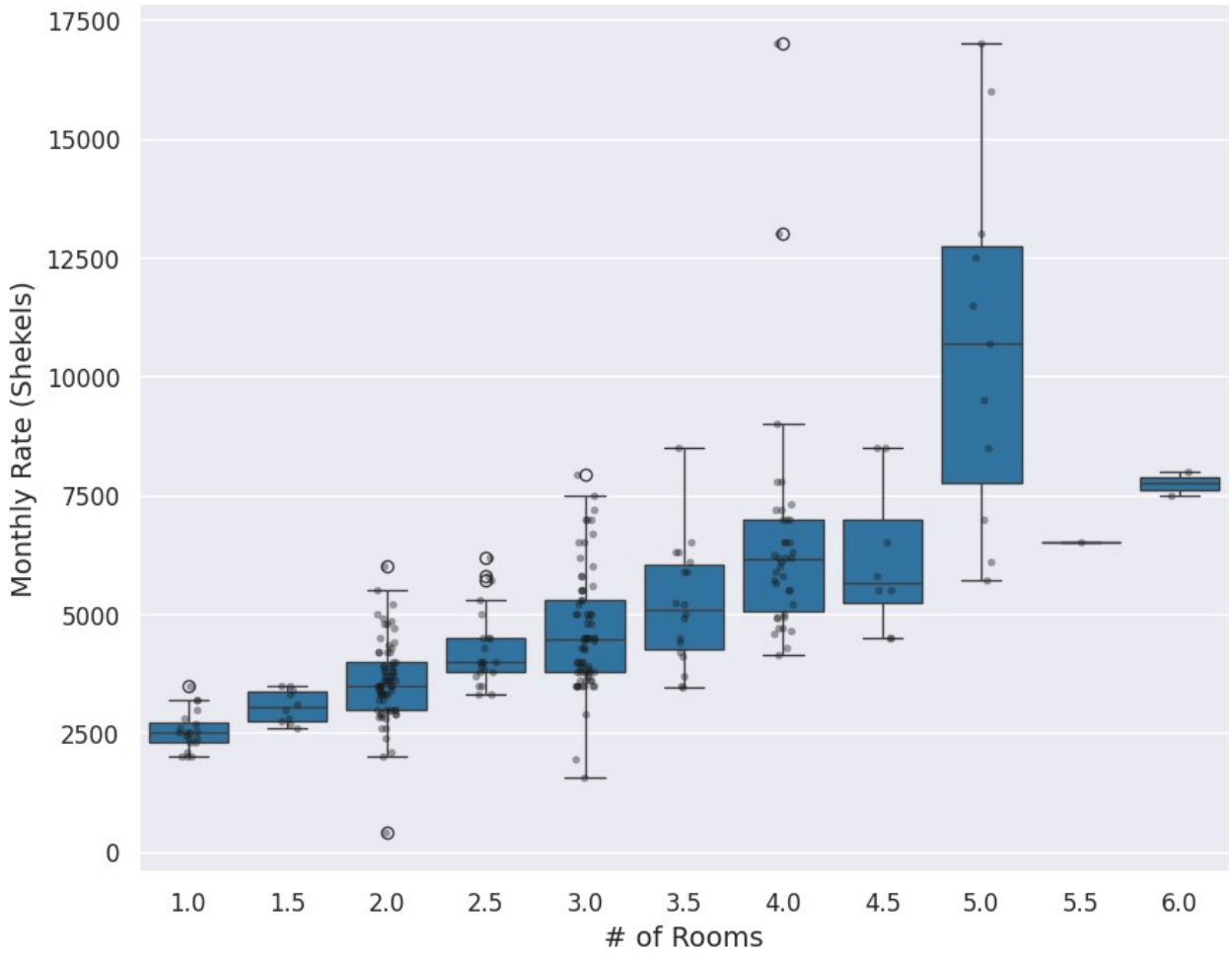
**Is there also a relation between the number of rooms and the listing price?**

Q: Create a visualization that compares the distribution of prices for different number of rooms. Your visualization should provide information about central tendency (mean/median/mode) and some information about the distribution of individual values around it (standard deviation/interquartile range) for each number of rooms. Also, show the real prices of the listings per number of rooms.

## Solution

```
# @title Solution
```

```
if clean_df_area_filtered is None:
  print("Can't run until 'clean_df_area_filtered' is created!")
else:
  plt.figure(figsize=(10,8))
  sns.boxplot(x='rooms', y='monthlyRate', data=clean_df_area_filtered, color='tab:blue
  sns.stripplot(x='rooms', y='monthlyRate', alpha=0.4 ,size=4,color='k',data=clean_df_
  plt.xlabel("# of Rooms")
  plt.ylabel("Monthly Rate (Shekels)");

  # Or:
  # plt.figure(figsize=(10,8))
  # sns.barplot(x='rooms', y='monthlyRate', data=clean_df_area_filtered, color='tab:bl
  # # Can also use mean but median is more informative in this case as prices are skew
  # sns.stripplot(x='rooms', y='monthlyRate', alpha=0.4 ,color='k',data=clean_df_area_
  # plt.xlabel("# of Rooms")
  # plt.ylabel("Monthly Rate (Shekels)");

  #Violin plot completly fails for very small subsets:
  # plt.figure(figsize=(10,8))
  # sns.violinplot(x='rooms', y='monthlyRate', data=clean_df_area_filtered, color='tab
  # plt.xlabel("# of Rooms")
  # plt.ylabel("Monthly Rate (Shekels)");
```



Now that we finished pre-processing the data, we can see the state of our outliers VS the data that remains:

```
if outlier_df is None:
  print("Can't run until 'outlier_df' is created!")
else:
  # describe the outlier data
  display(outlier_df.groupby('reason').describe())
  print(f"Proportion removed: {100*len(outlier_df) / (len(outlier_df)+len(clean_df_are
```

| | monthlyRate | | | | | | | | rooms |
|---|---|---|---|---|---|---|---|---|---|
| | count | mean | std | min | 25% | 50% | 75% | max | count |
| reason | | | | | | | | | |
| 'area' <= 10 | 5.0 | 5870.0 | 1399.821417 | 4000.0 | 5500.0 | 5500.0 | 6600.0 | 7750.0 | 5.0 |
| 'area' >= 800 | 2.0 | 5500.0 | 1555.634919 | 4400.0 | 4950.0 | 5500.0 | 6050.0 | 6600.0 | 2.0 |
| area > 800 | 2.0 | 5500.0 | 1555.634919 | 4400.0 | 4950.0 | 5500.0 | 6050.0 | 6600.0 | 2.0 |
| monthlyRate <= 0 | 25.0 | 0.0 | 0.000000 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 25.0 |

4 rows × 40 columns

Proportion removed: 11 %

# Submission Exercises

## Part 1: Diving deeper into rental prices

We will create a copy of the dataset and work on that. We want to make sure that we do not modify the original dataset.

### Part 1 - Create a DataFrame

```
# @title Part 1 - Create a DataFrame
part1_df = rent_df_backup_for_exercise.copy()
```

Let's go back to the distribution of monthly rental prices in the dataset. Are there interesting

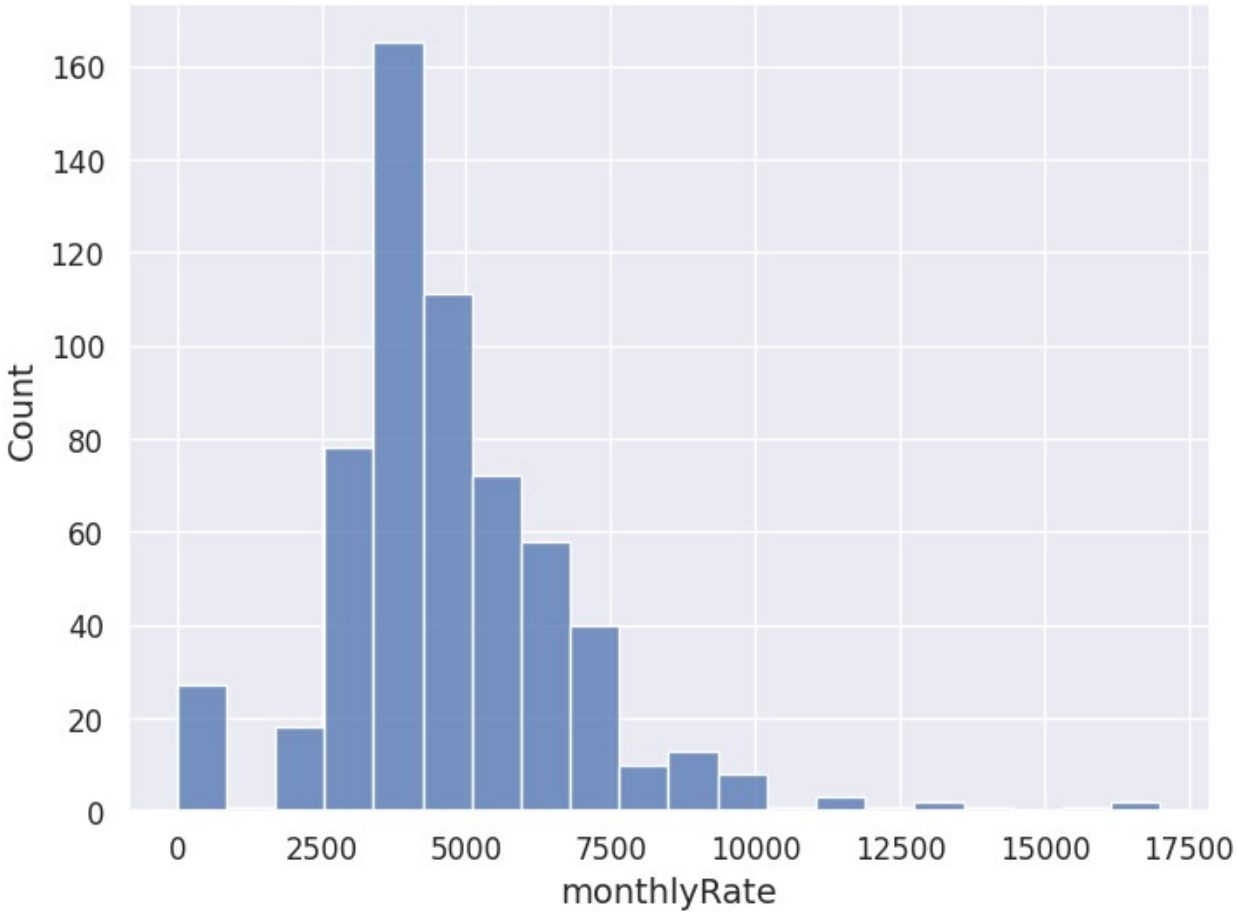
trends in the distribution that we missed in the visualizations before?

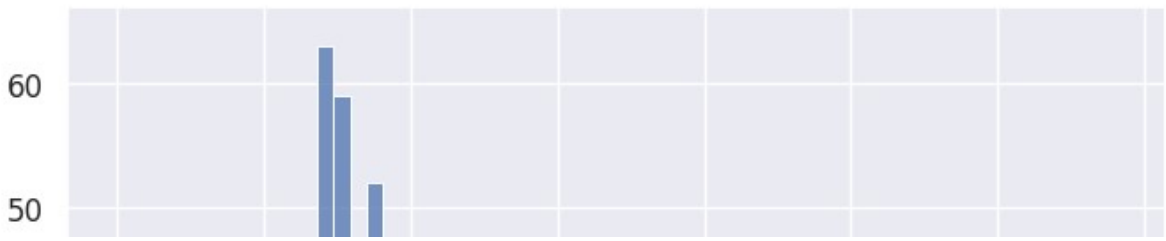
**Use only `part1_df` for the coding questions in this part**

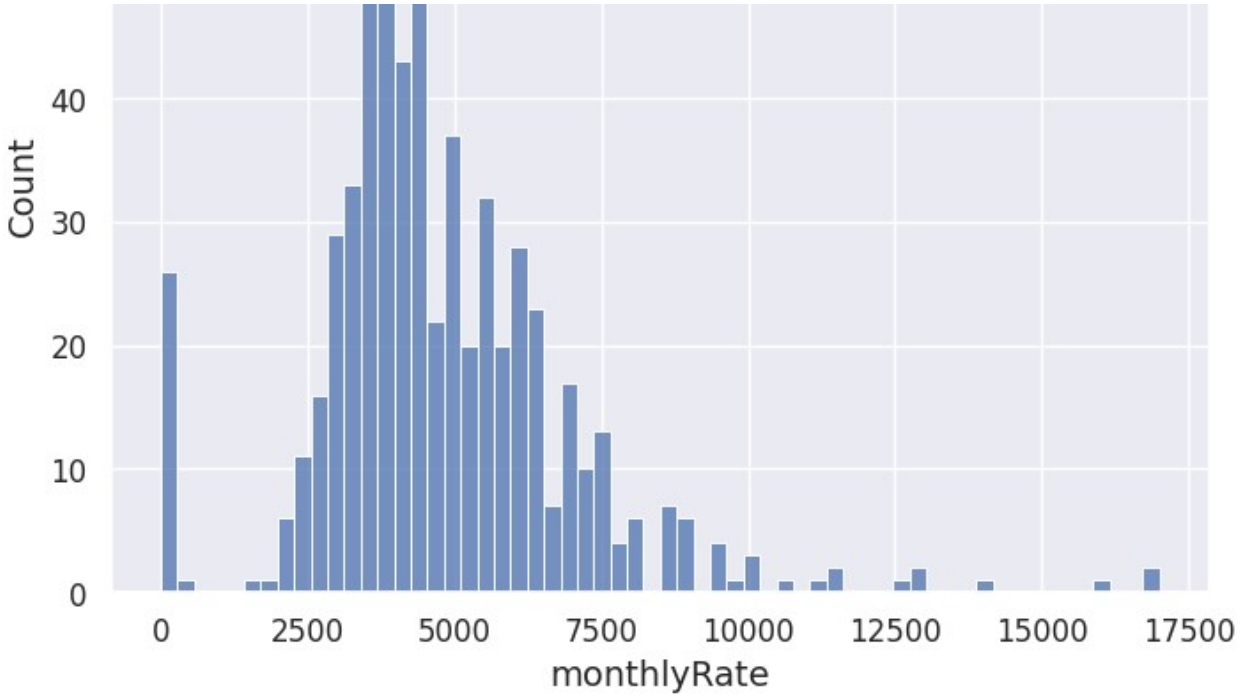
## Question 1

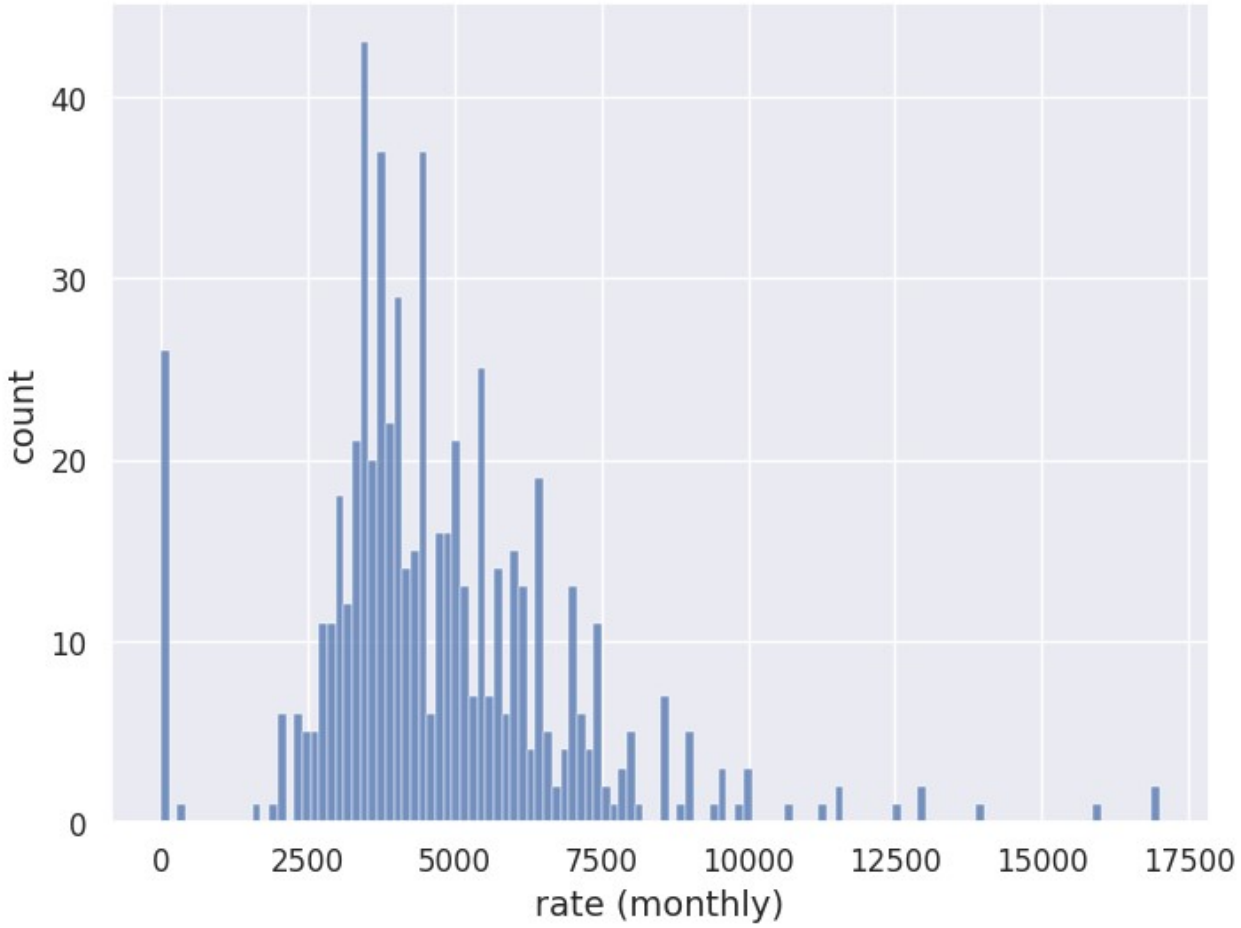
Plot 3 different histograms of the monthly prices with 20, 60 and 120 bins respectively, each in a different axis/figure.

```
fig = plt.figure(figsize=(8,6))
sns.histplot(x='monthlyRate', data=part1_df, bins=20)
fig = plt.figure(figsize=(8,6))
sns.histplot(x='monthlyRate', data=part1_df, bins=60)
fig = plt.figure(figsize=(8,6))
sns.histplot(x='monthlyRate', data=part1_df, bins=120)
plt.xlabel("rate (monthly)")
plt.ylabel("count");
```




Count
40
30
20
10
0
0
2500
5000
7500
10000
12500
15000
17500
monthlyRate


count
40
30
20
10
0
0
2500
5000
7500
10000
12500
15000
17500
rate (monthly)

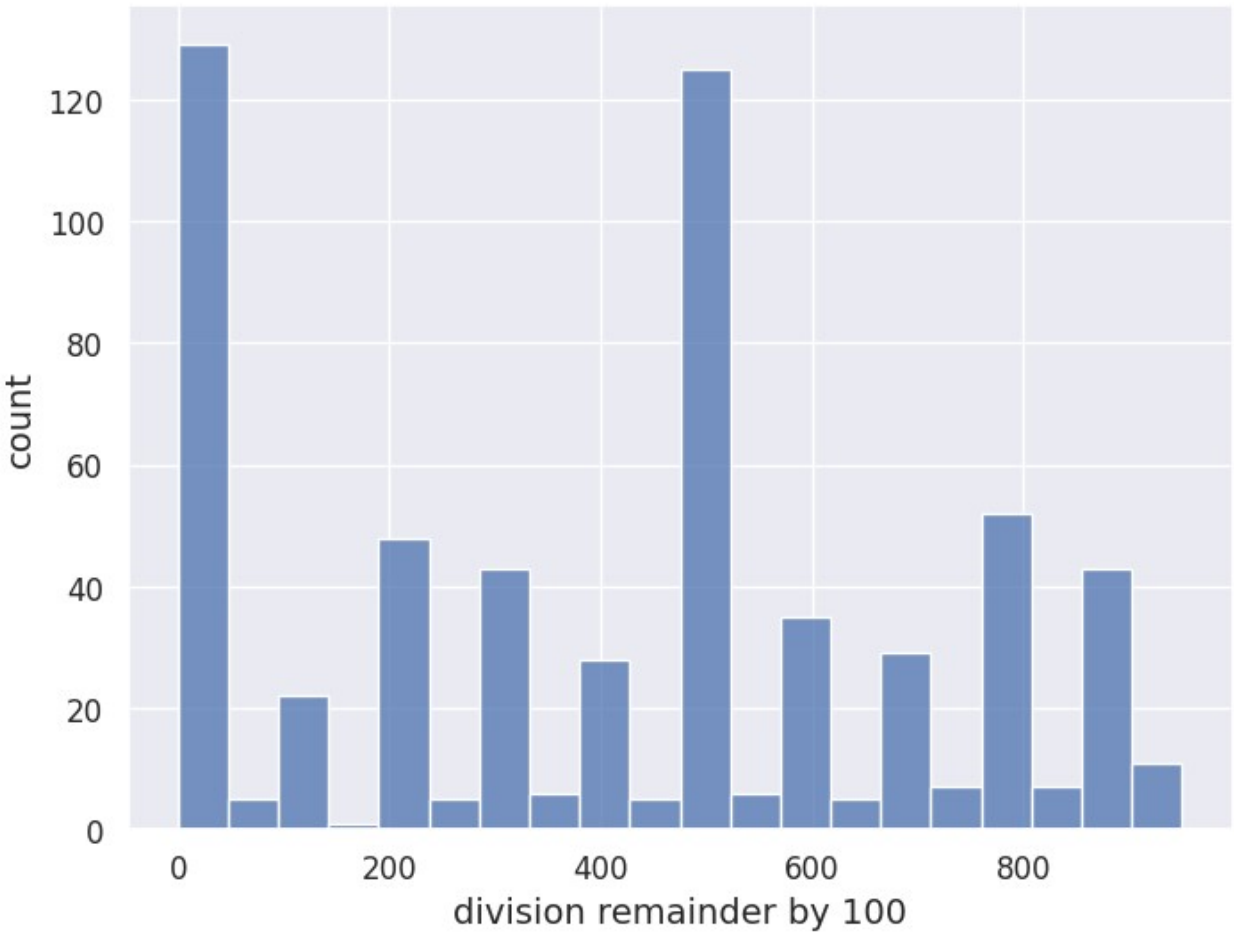
## Question 2

For 60 and 120 bins, you can see a repeating pattern of "peaks" and "vallies" in the distribution (mostly in the range between 500 and 7000). Is this pattern due to people rounding the rental prices? Please create a visualization that answers this question. Describe in words how the graph shows what the answer is (Hint: you can use the '%' operator to compute the remainder of dividing values in a pandas Series by a scalar number).

> **extra hint**: please open this cell only after discussing with the course staff the best solution you could come up with

Show code

```
part1_df["division remainder"] = part1_df["monthlyRate"]%1000
fig = plt.figure(figsize=(8,6))
sns.histplot(x='division remainder', data=part1_df, bins=20)
plt.xlabel("division remainder by 100")
plt.ylabel("count");
```



---

Part 1 Question 2 - textual Answer:

בדקתי על כל התצפיות מה תהיה שארית החלוקה כאשר נחלק את כל המחירים באלף. המסקנה שהגעתי אליה היא שרוב האנשים מעגלים בקפיצות של 500, אחר כך הרבה מעגלים לפי 100 ומעטים לפי 50.

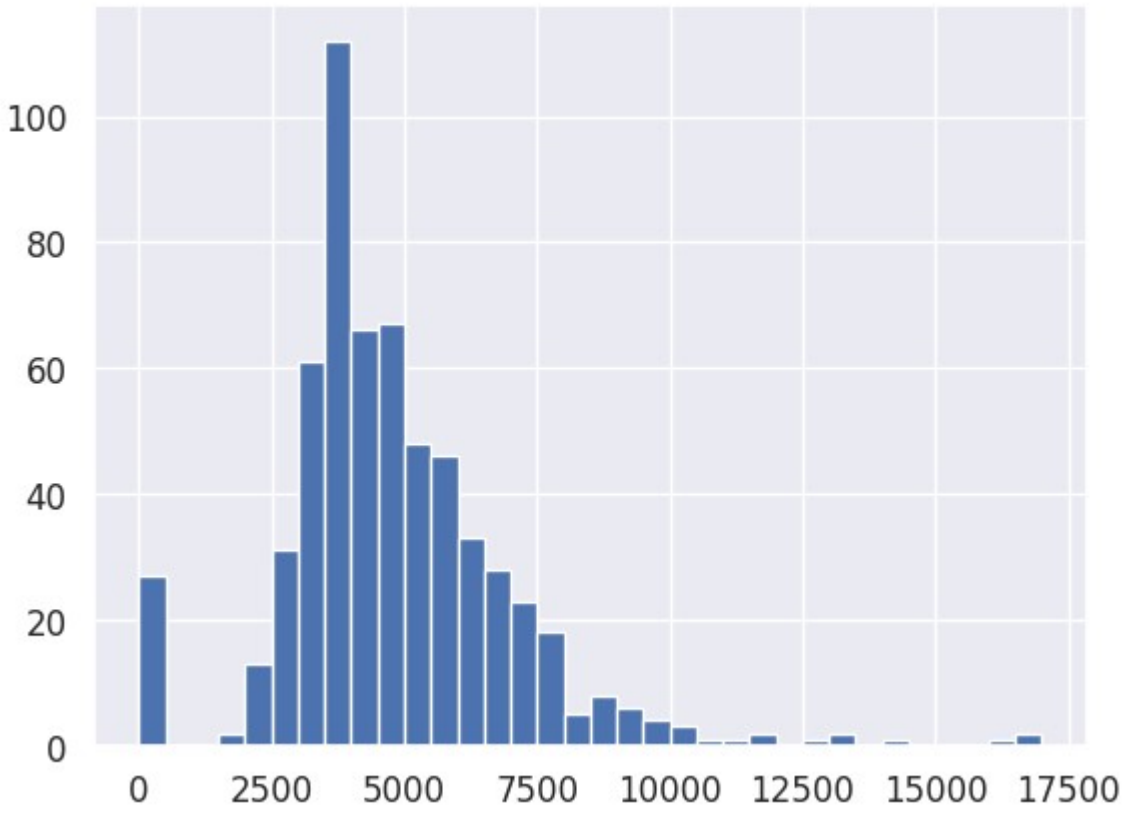
---

## Question 3

We expect to see a "drop" in prices frequency near the 5000 Shekels mark due to tax considerations (See here for an explanation). Create a histogram visualization of the data with the smallest possible bins such that every bin will include exactly one multiplication of 500 (Hint: read the `bins` parameter documentation and what types it accepts). Explain why does this choice of bin size ensures that we will not see rounding effects. Do you see a "drop" around 5000 Shekels? Are there other "drops"?

```
part1_df["monthlyRate"].hist(bins=np.arange(0, max(part1_df["monthlyRate"]) + 500, 500
```

```
<Axes: >
```




---

Part 1 Question 3 - textual Answer:

אנשים מעגלים לרוב לפי 500 לכן כאשר אנחנו עושים קפיצות של 500 אנחנו מדוואים שהעמודות לא שכיחות יותר בגלל הנטיה לעגל. ניתן לראות כי יש ירידה חדה אחרי שעוברים את ה-5000. בנוסף, הוא בשכיחות מאוד גבוהה כך שניתן להסיק כי אנשים שרוצים מעט יותר מ5000 מגבילים את עצמם ולכן יש שם הרבה תצפיות.

---

# Part 2: Size or number of rooms?

## Part 2 - Create a DataFrame for Part 2

```
# @title Part 2 - Create a DataFrame for Part 2

# Create the dataframe and remove the outliers we found in the intro part:
part2_df = rent_df_backup_for_exercise.copy()
part2_df = part2_df[part2_df['monthlyRate'] > 0].reset_index(drop=True);
part2_df = part2_df[part2_df['area'] < 800].reset_index(drop=True)
part2_df = part2_df[part2_df['area'] > 10].reset_index(drop=True)
```

We saw that both the number of rooms and the area of an apartment are strongly associated with the monthly rate. We now want to check if those are just two perspectives of the same relation (how big is the apartment) or is there something more to it. We will use the cleaned dataframe for this exercise.

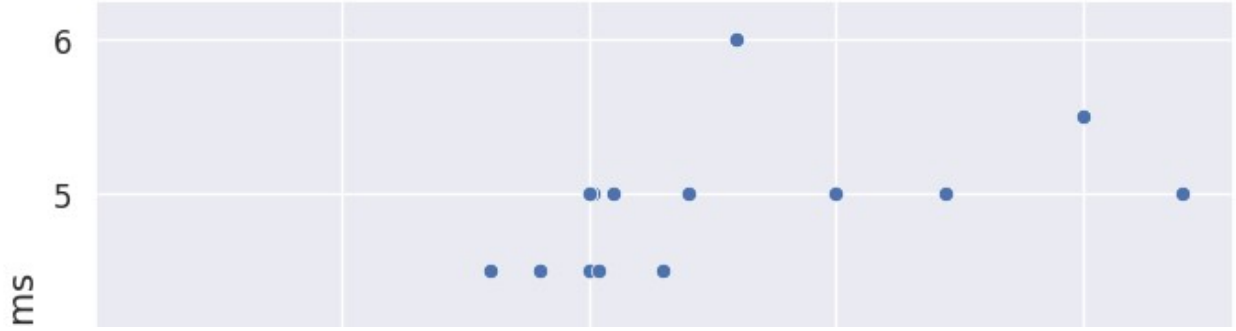
**Use only `part2_df` for the coding questions in this part**

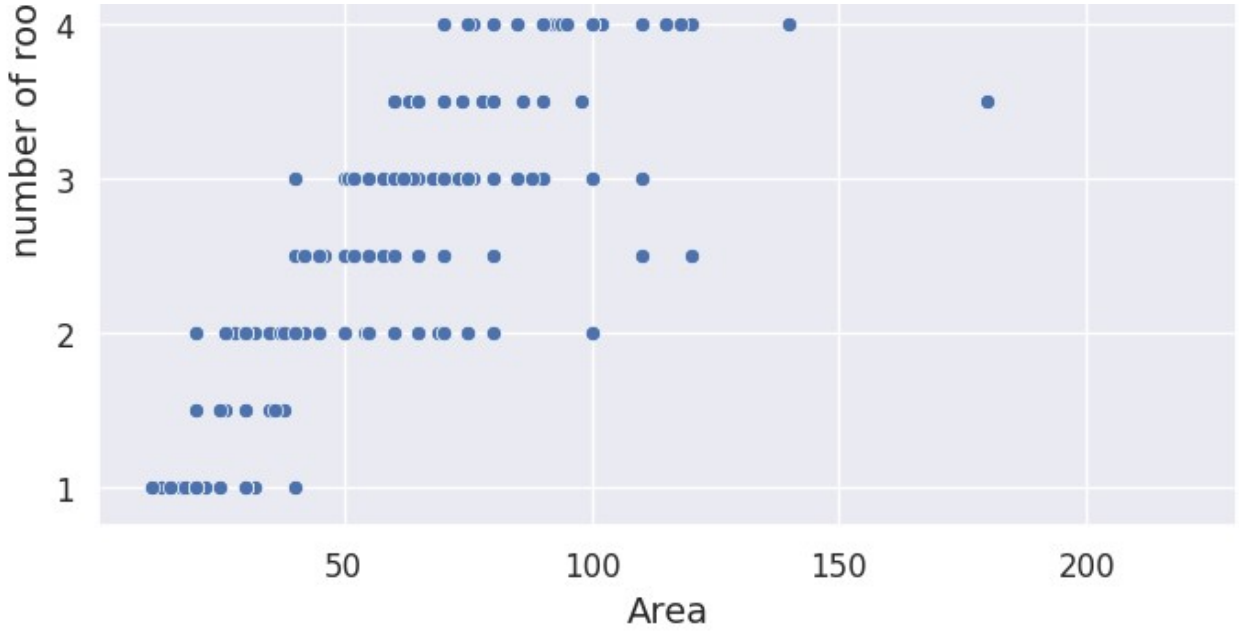
## Question 1

Generate a visualization to show that there is a strong association between the number of rooms and the area of the apartment. Explain your choice of plot type and your conclusion from the graph.

```
# Part 2 - Question 1
plt.figure(figsize=(8,6))
sns.scatterplot(x='area', y='rooms', data=part2_df)
plt.xlabel("Area")
plt.ylabel("number of rooms")
```

```
Text(0, 0.5, 'number of rooms')
```






---

Part 2 Question 1 - textual Answer:

בחרנו בגרף נקודות בגלל שרצינו לראות קוראלציות. נראה שיש קורלאציה חיובית חזרה בין שטח הדירה לכמות החדרים בה. נראה שהקוראלציה נחלשת ככל ששטח הדירה גדול יותר.

---

## Question 2

Add a new column to the dataframe named `"averageRoomSize"` with the average room size in the given listing.

```
# Part 2 - Question 2
part2_df["averageRoomSize"] = (part2_df["area"] / part2_df["rooms"])
print(part2_df)
```

```
     propertyID neighborhood  monthlyRate mefarsem  rooms  floor   area  \
0       3994505   2000.0       קריית יובל  private    1.0    2.0   13.0
1       3981298        2450.0       רחביה  private    1.0    1.0   17.0
2       3981623         2550.0       מלחה  private    1.0    0.0   30.0
3       3993997      2100.0       בית וגן  private    1.0    0.0   15.0
4       3994399     2300.0       פסגת זאב  private    1.0    1.0   32.0
..          ...          ...          ...      ...    ...    ...    ...
268     3986876     5700.0       רמת שלמה  private    5.0    3.0  100.0
269     3993009         10700.0      רמות  private    5.0    0.0  150.0
270     3981999       13000.0      תלפיות  private    5.0    4.0  150.0
271     3994215   2900.0       קריית יובל  private    2.0    0.0   40.0
272     3972927  4900.0       קריית שמואל    agent    2.0    0.0   55.0

          entry                                        description  numFloors  \
0    10/08/2022  2.0        יחידת דיור להשכרה ברחוב הראשי של קריית יובל, ה...
```

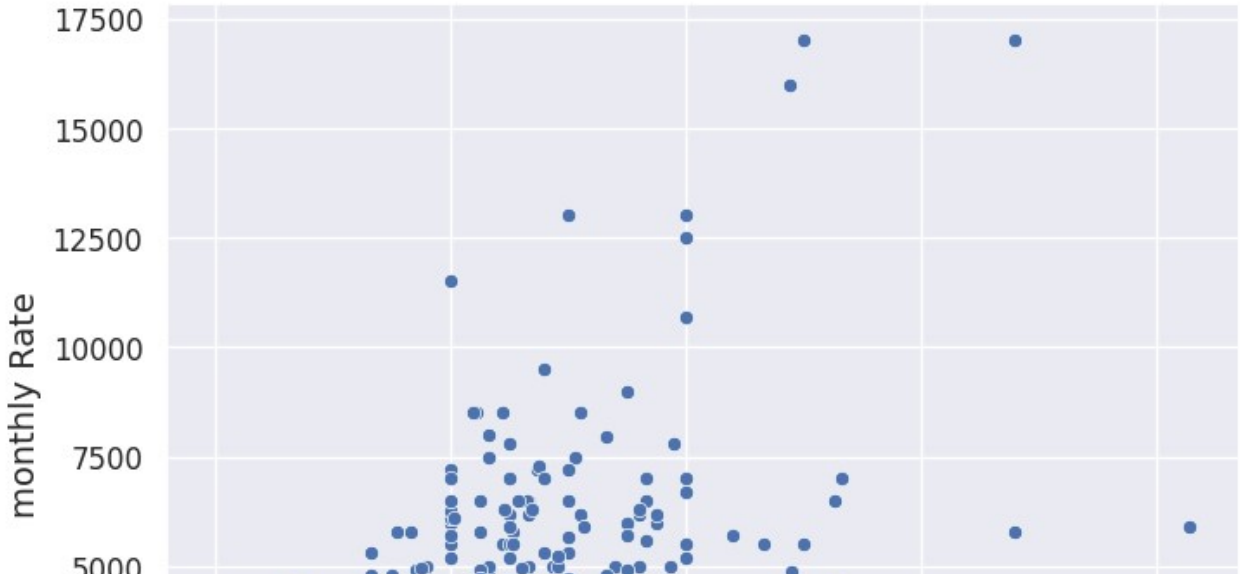
```
1    10/08/2022                3.0       דירת יחיד 17 מטר כולל מרפסת קטנה
2    10/08/2022  2.0        ... דירה יפה ומטופחת, לדיירת שקטה לטווח ארוך, ללא
3    10/08/2022  3.0       ...דירת חדר, כ-15 מ"ר, במיקום מרכזי אך שקט, משופצ
4    10/08/2022  1.0       ...בס"ד  בפסגת זאב מזרח דירת חדר גדולה משופצת ויפ
..          ...                                      ...       ...
268  10/08/2022  4.0       ...להשכרה, דירה, קומה 3, בירושלים וגם בקומה 4 דיר
269  10/08/2022  4.0       ... דירה שמורה ומטופחת עם כניסה פרטית ללא דמי ועד
270  10/08/2022  5.0       ...דירת 5 חדרים חדשה! בדירה יש מרפסת  מרפסת שירו
271  10/08/2022                                   NaN       4.0
272  10/08/2022  3.0       ... דירה מרווחת ומשופצת , סלון גדול ומרפסת מתאימה

     averageRoomSize
0               13.0
1               17.0
2               30.0
3               15.0
4               32.0
..               ...
268             20.0
269             30.0
270             30.0
271             20.0
272             27.5

[273 rows x 11 columns]
```

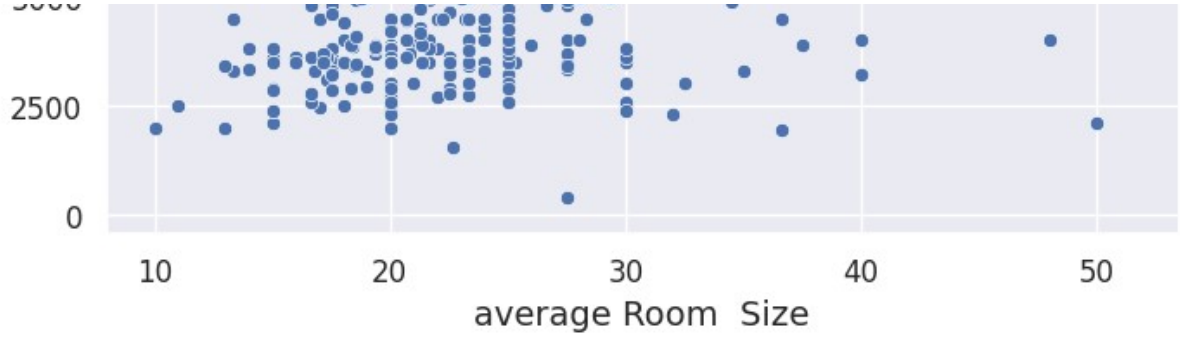
## Question 3

Create a plot of the relation between the average room size and the monthly rate.

```
# Part 2 - Question 3
plt.figure(figsize=(8,6))
sns.scatterplot(x='averageRoomSize', y='monthlyRate', data=part2_df)
plt.xlabel("average Room  Size")
plt.ylabel("monthly Rate")
```

```
Text(0, 0.5, 'monthly Rate')
```





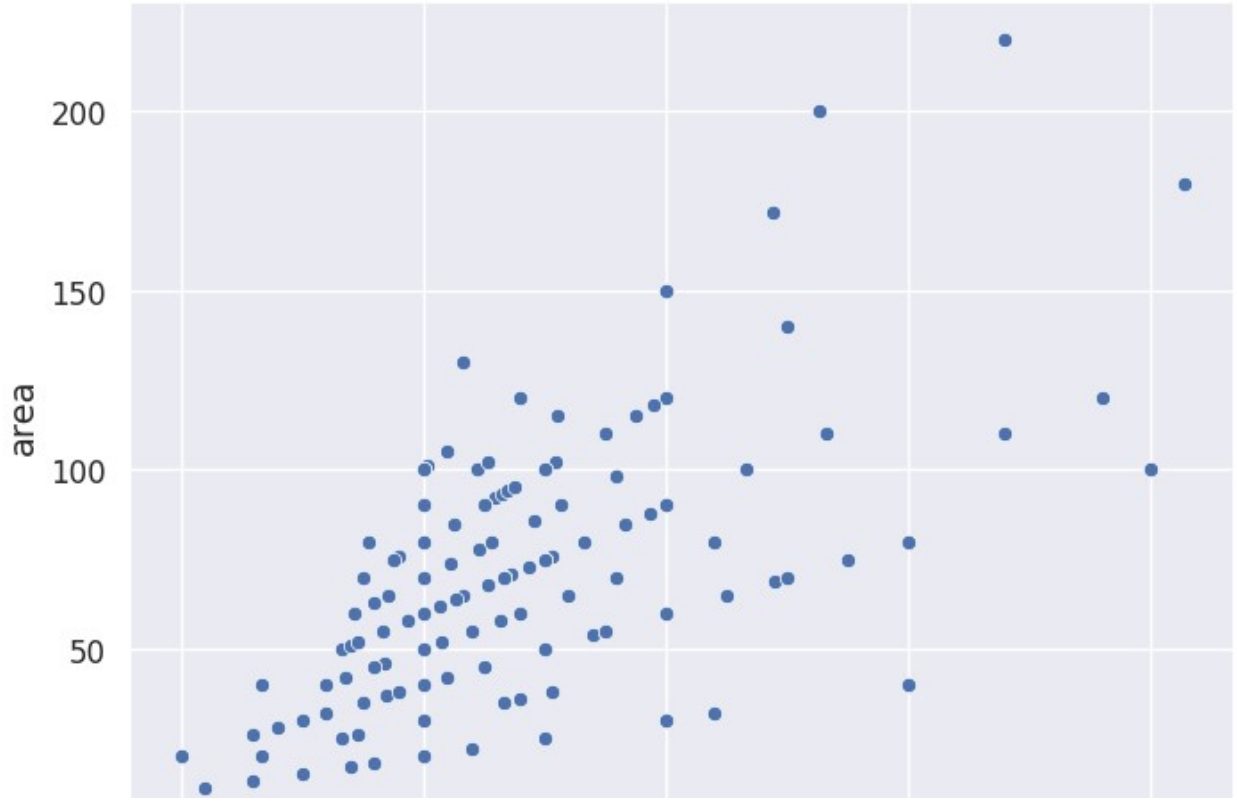

## Question 4 - **bonus**

We can see that the variance of the monthly rate increases with the average room size.

Suggest what might be the reason for the increase in the variance and create a visualization to support or refute your suggestion.

```
# Part 2 - Question 4
plt.figure(figsize=(8,6))
sns.scatterplot(x='averageRoomSize', y='area', data=part2_df)
plt.xlabel("average Room  Size")
plt.ylabel("area")
```

```
Text(0, 0.5, 'area')
```




---

Part 2 Question 4 - textual Answer:

ניתן לראות שככל שגודל החדרים עולה רואים יותר שונות בגודל הדירה בכללי. לכן, נראה גם יותר שונות במחירי הדירות.

---

# Part 3: Neighborhoods

## Part 3 - Function Definitions and DataFrame Creation

```
# @title Part 3 - Function Definitions and DataFrame Creation
def reverse_string(a):
  return a[::-1]


socialrank_df = load_df(SOCIORANK_ID)
neighborhood_ranks = {k: v for k,v in zip(socialrank_df['neighborhood'], socialrank_df

def get_neighborhood_rank(neighborhood):
  if neighborhood in neighborhood_ranks:
    return neighborhood_ranks[neighborhood]
  else:
    return None

# Create the dataframe and remove the outliers we found in the intro part:
part3_df = rent_df_backup_for_exercise.copy()
part3_df = part3_df[part3_df['monthlyRate'] > 0].reset_index(drop=True);
part3_df = part3_df[part3_df['area'] < 800].reset_index(drop=True)
part3_df = part3_df[part3_df['area'] > 10].reset_index(drop=True)
part3_df["neighborhood_flipped"] = part3_df["neighborhood"].apply(reverse_string) # ma
```

We now want to focus on the differences between different neighborhoods in Jerusalem.

**Use only `part3_df` for the coding questions in this part**

*Use the `"neighborhood_flipped"` column for visualizations as seaborn will flip the order of letters in hebrew.

## Question 1

Print the number of unique neighborhoods that appear in the dataset.

```
# Part 3 - Question 1
print(len(part3_df["neighborhood_flipped"].unique()))
```

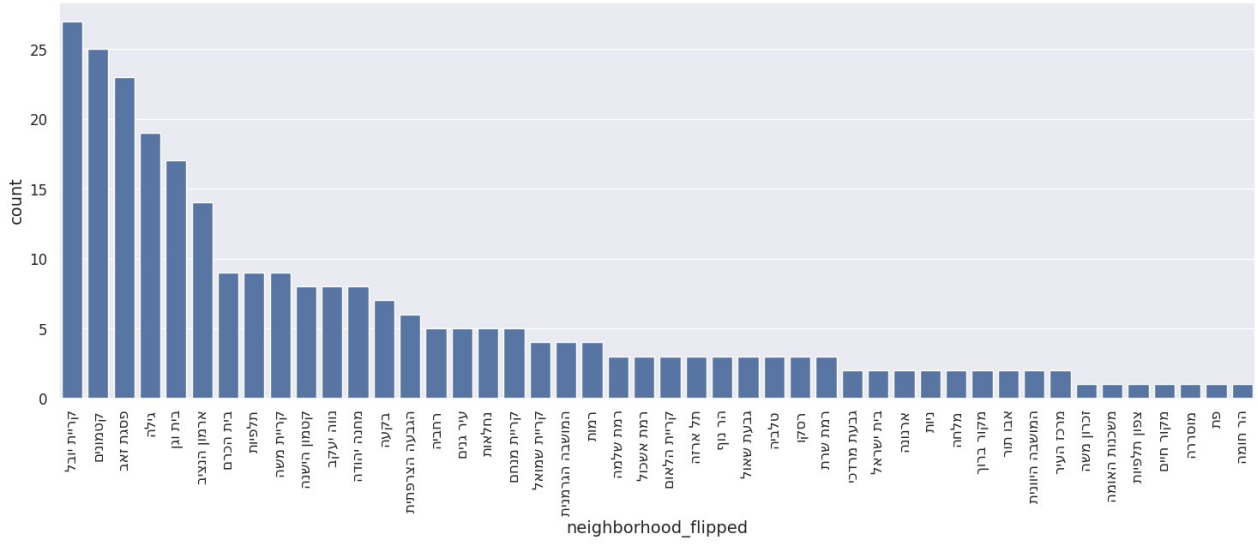
```
46
```

## Question 2

Visualize the number of listings per neighborhood in a way that will allow you to easily identify those with the highest count.

```
sorted = list(part3_df['neighborhood_flipped'].value_counts(sort=True).keys())
plt.figure(figsize=(18,6))
plt.xticks(rotation=90)
sns.countplot(x='neighborhood_flipped', order=sorted, data=part3_df)
```

```
<Axes: xlabel='neighborhood_flipped', ylabel='count'>
```



## **Question 3** - Heavy-tailed distributions

Print the number of neighborhoods with less than 5 listings and the fraction of their total number of listings out of the total number of listings. Also print the fraction of listings from the 8 most frequent neighborhoods out of the total number of listings.

```
# Part 3 - Question 3
part3_df["number of listing"] = part3_df['neighborhood'].map(part3_df['neighborhood'].
print("number of neighbrhoods with less then 5 listings")
print(len(part3_df[part3_df['number of listing'] < 5]['neighborhood'].unique()))
print("fraction of neighbrhoods with less then 5 listings")
print(part3_df[part3_df['number of listing'] < 5]["number of listing"].sum() / part3_d
print("fraction of the 8 most frequent neighbrhoods")
most_frequent_neighborhoods = part3_df['neighborhood'].value_counts().head(8).index
print(part3_df[part3_df['neighborhood'].isin(most_frequent_neighborhoods)]['number of
```

```
number of neighbrhoods with less then 5 listings
28
fraction of neighbrhoods with less then 5 listings
0.048849758591309286
fraction of the 8 most frequent neighbrhoods
0.8210735586481114
```

Those types of distributions where there are many categories that appear only a few times but together take a large portion of the distribution are called heavy-tailed (or long-tailed) distributions. This is a real issue in many data science applications, since even if we have a large dataset there are still some sub-populations or sub-categories that are not well represented.

## **Question 4**

Create a new filtered dataframe with listings from only the 8 most frequent neighborhoods.

```
# Part 3 - Question 4
most_frequent_neighborhoods = part3_df['neighborhood'].value_counts().head(8).index
only8_df = part3_df[part3_df['neighborhood'].isin(most_frequent_neighborhoods)]
print(only8_df)
```

```
      propertyID neighborhood  monthlyRate mefarsem  rooms  floor   area  \
0        3994505   2000.0       קריית יובל  private    1.0    2.0   13.0
```

```
3       3993997      2100.0       בית וגן  private    1.0    0.0   15.0
4       3994399     2300.0       פסגת זאב  private    1.0    1.0   32.0
11      3986231   2600.0       קריית יובל  private    1.0    1.0   30.0
14      3992479   2400.0       קריית יובל  private    1.0    1.0   20.0
..          ...          ...          ...     ...    ...    ...    ...
262     3988577     6500.0       פסגת זאב  private    5.5    1.0  200.0
264     3993965      7000.0       בית וגן  private    5.0    3.0  120.0
266     3974914       17000.0      תלפיות  private    5.0    3.0  220.0
270     3981999       13000.0      תלפיות  private    5.0    4.0  150.0
271     3994215   2900.0       קריית יובל  private    2.0    0.0   40.0

          entry                                        description  numFloors  \
0    10/08/2022  2.0        יחידת דיור להשכרה ברחוב הראשי של קריית יובל, ה...
3    10/08/2022  3.0        דירת חדר, כ-15 מ"ר, במיקום מרכזי אך שקט, משופצ...
4    10/08/2022  1.0        בס"ד  בפסגת זאב מזרח דירת חדר גדולה משופצת ויפ...
11   10/08/2022  2.0        הדירה שטופת שמש,  מגיעה מרוהטת- מיטה, ארון בגד...
14   10/08/2022  1.0        דירת חדר חמודה עם גינה קטנה משותפת,מתאים ליחיד...
..          ...                                                ...        ...
262  10/08/2022  3.0        דירה בת 5.5 חדרים . בקומה התחתונה סלון , מטבח ...
264  10/08/2022  3.0        דירת 5 חדרים ובעלת 5 מרפסות קטנות, עברה צביעה ...
266  10/08/2022  4.0        דירת 5 חדרים ענקית ומהממת, בבנין בוטיק ויחודי ...
270  10/08/2022  5.0        דירת 5 חדרים חדשה! בדירה יש מרפסת  מרפסת שירו...
271  10/08/2022                                             NaN        4.0

    neighborhood_flipped  number of listing
0             27                 לבוי תייק
3                17                וגו תיב
4               23                באז תגספ
11            27                 לבוי תייק
14            27                 לבוי תייק
..                ...                ...
262             23                באז תגספ
264              17                וגו תיב
266               9                 תויפלת
270               9                 תויפלת
271           27                 לבוי תייק

[143 rows x 12 columns]
```

## Question 5

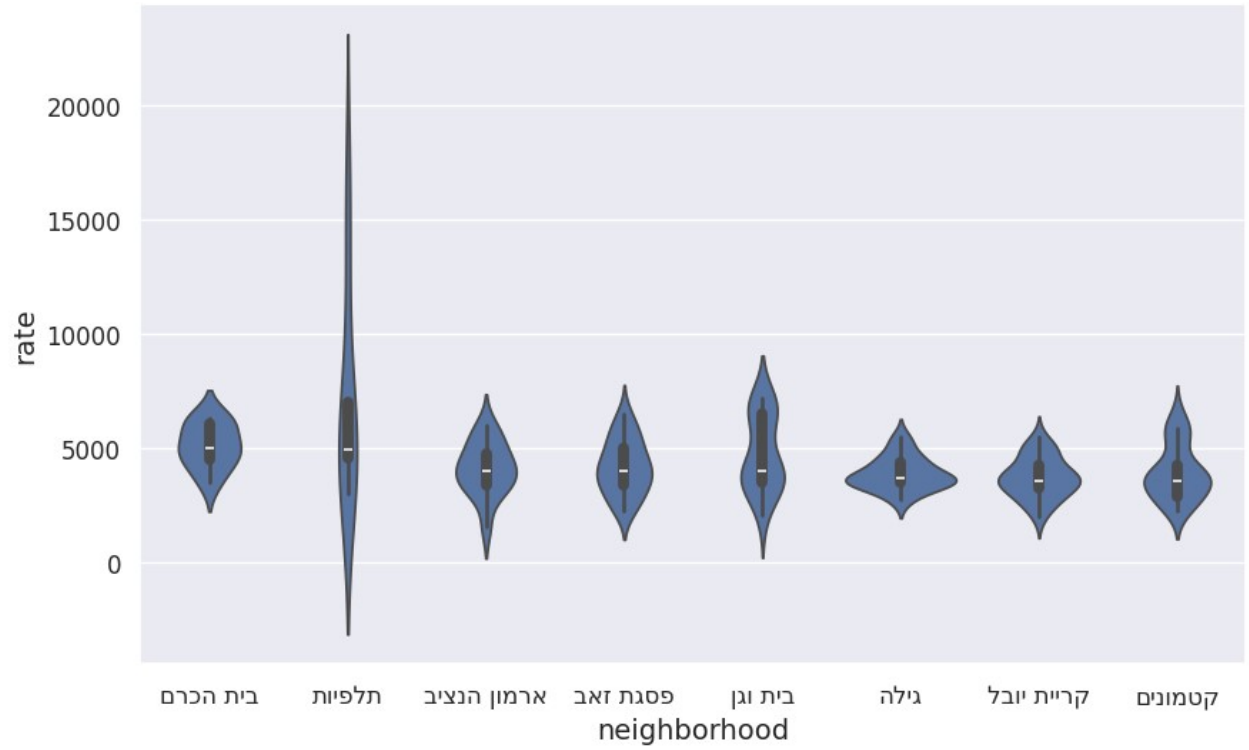
Plot a graph to check whether there are different distributions of monthly rates in the eight neighborhoods. Explain your choice for the visualization and your conclusions. Note: Make sure that the neighborhoods are ordered in the plot based on their tendency for higher or lower monthly rates.

Hint: Which is a better descriptor of the central tendency of monthly rates when the distributions are skewed?

```
# Part 3 - Question 5
most_frequent_neighborhoods = part3_df['neighborhood_flipped'].value_counts().head(8).
only8_df = part3_df[part3_df['neighborhood_flipped'].isin(most_frequent_neighborhoods)
median_rates = only8_df.groupby('neighborhood_flipped')['monthlyRate'].median()
sorted_neighborhoods = median_rates.sort_values(ascending=False).index
```

```
plt.figure(figsize=(10, 6))
sns.violinplot(x='neighborhood_flipped', y='monthlyRate', data=only8_df, order=sorted_
plt.xlabel("neighborhood")
plt.ylabel("rate")
```

```
Text(0, 0.5, 'rate')
```




---

Part 3 Question 5 - textual Answer:

בחרנו בגרף כינור בגלל שהוא מציג לנו אתכל התצפיות וההתפלגויות שלהן. ככה אנחנו יכולים לדעת את האזורים השכיחים יותר ואת תצפיות הקיצון. את התצפיות סידרנו לפי הסדר של החציון מכיוון שהחציון אינו מוטה בעקבות תצפיות קיצון וניתן לראות כי החציון של כל התצפיות יחסית שווה.

---

## Question 6

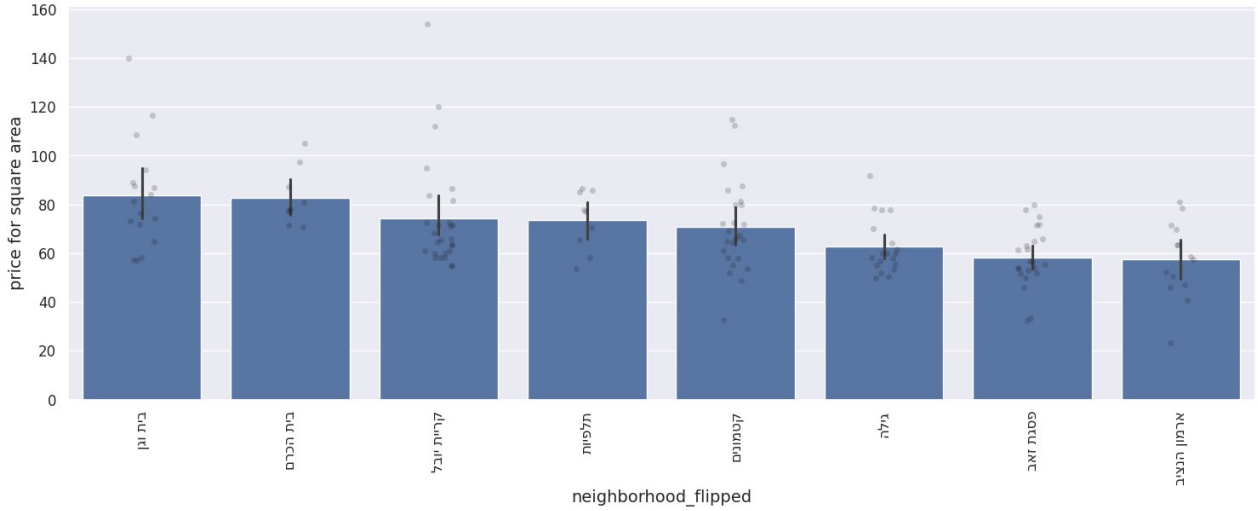
Now that we compared the different distributions of monthly rates betwen neighborhoods, we can check whether we can explain some of the differences using our common-sense and the data we already have. For example, perhaps different neighborhoods have different distributions of apartment sizes?

distributions of apartment sizes?

Think of a new variable that will allow you to check the relationship between neighborhoods and prices fairly, factoring different apartment sizes out of the equation. Save this measure into the dataframe and create a new visualization to answer the question.

```
# Part 3 - Question 6
only8_df["price for square area"] = (only8_df["monthlyRate"]/only8_df["area"])
median_price_per_area = only8_df.groupby('neighborhood_flipped')['price for square are
sorted_neighborhoods = median_price_per_area.sort_values(ascending=False).index
plt.figure(figsize=(18,6))
plt.xticks(rotation=90)
sns.barplot(x='neighborhood_flipped', y='price for square area', order=sorted_neighbor
sns.stripplot(x='neighborhood_flipped', y='price for square area', order=sorted_neighb
plt.xlabel("neighborhood_flipped")
plt.ylabel("price for square area")
```

```
<ipython-input-88-353acf99dab4>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable
  only8_df["price for square area"] = (only8_df["monthlyRate"]/only8_df["area"])
Text(0, 0.5, 'price for square area')
```



---

## Part 3 Question 6 - textual Answer:

בכדי לבחון האם יש שכונות יקרות יותר מאחרות ולנטרל את גודל הדירה בחנו את המחיר לכל דירה באמצעות מחיר למטר רבוע. לאחר מכן הצגנו את הממצעים בגרף עמודות ממוין לפי הממוצע והצגנו את התצפיות. נראה כי יש הבדלים משמעותיים בין השכונות היקרות ביותר לשכונות הכי פחות יקרות

---

Given the conclusions from the previous steps, we may think that the apartment's neighborhood gives us additional information about the expected monthly rate. But the sample size for most neighborhoods is rather small. So let's examine another way to utilize the location information. Luckily, we also have data about the socio-economic rank of most neighborhoods (between 1 and 10).

## Question 7 - **bonus**

Use again the full dataset (without filtering by neighborhood).

Create an aggregated dataframe where every record represents a neighborhood, with columns for:

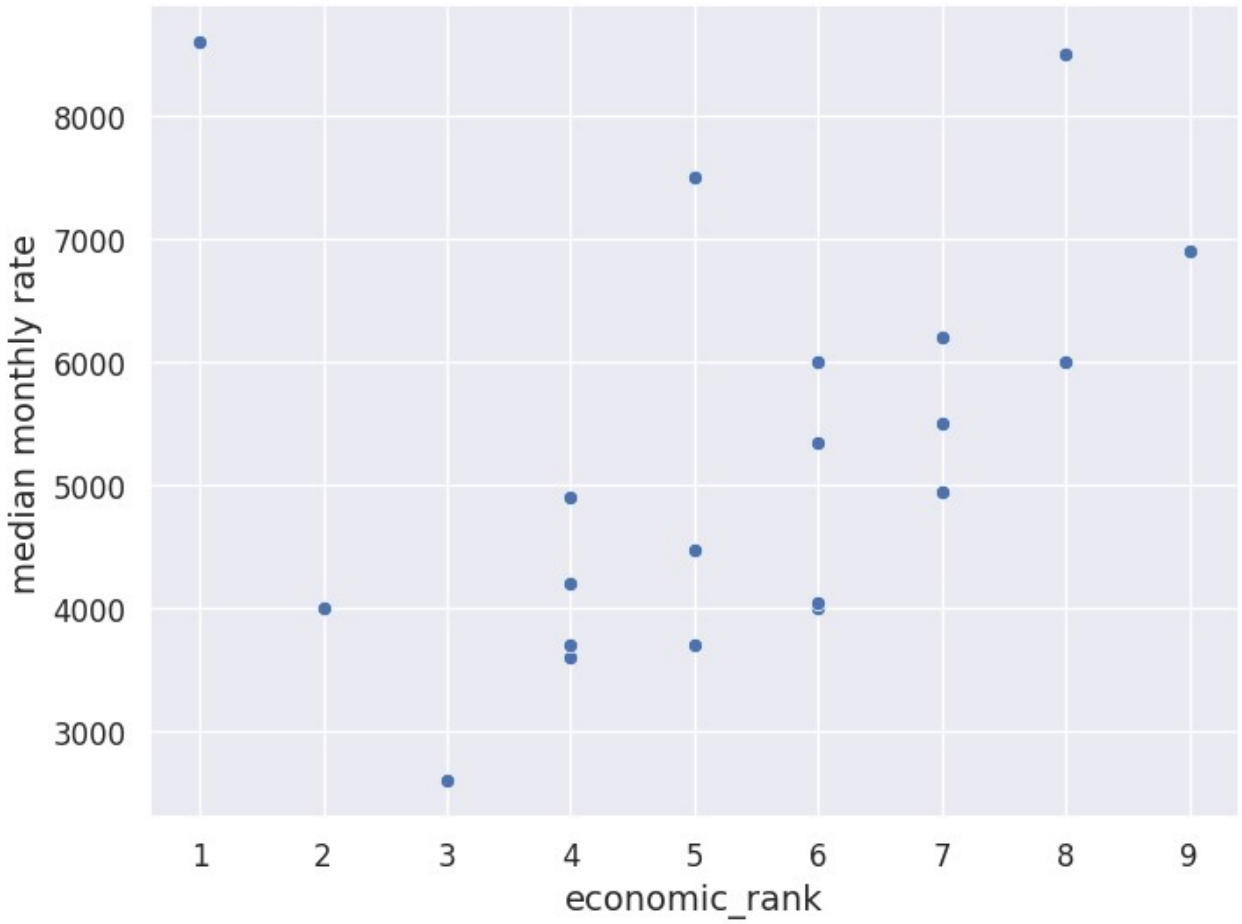
1. neighborhood name
2. flipped neighborhood name
3. The number of listings in a neighborhood
4. The median monthly rate for listings in this neighborhood.

Add a column with the neighborhood socio-economic rank to the dataframe (you can use the provided `get_neighborhood_rank` function that takes as an input a neighborhood name and returns its socio-economic rank.) Use this dataframe to visualize the association between socio-economic rank and pricing for all neighborhoods with at least 5 listings. What is you conclusion?

```
# Part 3 - Question 7
part3q7_df = pd.DataFrame({'neighborhood': part3_df['neighborhood'].unique(),'neighbor
part3q7_df["listing_num"] = part3_df['neighborhood'].map(part3_df['neighborhood'].valu
part3q7_df["median monthly rate"] = part3q7_df['neighborhood'].map(part3_df.groupby('n
part3q7_df["economic_rank"] = part3q7_df['neighborhood'].apply(get_neighborhood_rank)

part3q7_df = part3q7_df[part3q7_df["listing_num"] >= 5]

plt.figure(figsize=(8,6))
```

```
plt.figure(figsize=(8,6))
sns.scatterplot(x='economic_rank', y='median monthly rate', data=part3q7_df)
plt.xlabel("economic_rank")
plt.ylabel("median monthly rate")
```

```
Text(0, 0.5, 'median monthly rate')
```




---

Part 3 Question 7 - textual Answer:

נראה כי יש מתאם חיובי בין הרמה הסוציואקונוממית של השכונה לבין המחיר החציוני לחודש.

## Part 4: Are private houses more expensive than apartments?

### Part 4 - Create a DataFrame and remove outliers for Part 4

```
# @title Part 4 - Create a DataFrame and remove outliers for Part 4
```

```
# @title Part 4 - Create a DataFrame and remove outliers for Part 4
part4_df = rent_df_backup_for_exercise.copy()
part4_df = part4_df[part4_df['monthlyRate'] > 0].reset_index(drop=True);
part4_df = part4_df[part4_df['area'] < 800].reset_index(drop=True)
part4_df = part4_df[part4_df['area'] > 10].reset_index(drop=True)
```

Finally, we want to check if listings in private houses tend to be more expensive than apartments in a building.

**Use only `part4_df` for the coding questions in this part**

## Question 1

The current dataset doesn't include a variable that describes whether a listing is in a building or a private house but this can be inferred from the existing variables. Create a new column named "is_a_house" with value of `True` if a listing is in the first (or zero) floor in a building with only one floor. Print the number of private houses and print the descriptions of three random listings with 'is_a_house' equal to `True`.

```
# Part 4 - Question 1
part4_df["is_a_house"] = False
part4_df.loc[(part4_df['numFloors'] == 1) & ((part4_df['floor'] == 0) | (part4_df['floor
print(len(part4_df["is_a_house"][part4_df["is_a_house"] == True]))
print(part4_df[part4_df["is_a_house"] == True].sample(3))
```

```
17
     propertyID   neighborhood  monthlyRate mefarsem  rooms  floor   area  \
92      3882274       2400.0       פסגת זאב  private    2.0    1.0   30.0
171     3982071  7000.0       הגבעה הצרפתית  private    3.0    0.0  110.0
48      3985356           3600.0       גילה  private    2.0    1.0   60.0

          entry                                        description  numFloors  \
92   10/08/2022  1.0        ...יחידת דיור מוארת, משופצת . כניסה פרטית! דירה
171  10/08/2022  1.0        ...דירה ברחוב שקט חנייה בשפע כניסה פרטית , מרפסת
48   10/08/2022  1.0        ...מקום מדהים ,קומה ראשונה,תחנת אוטובוס,גני ילדים

     is_a_house
92         True
171        True
48         True
```

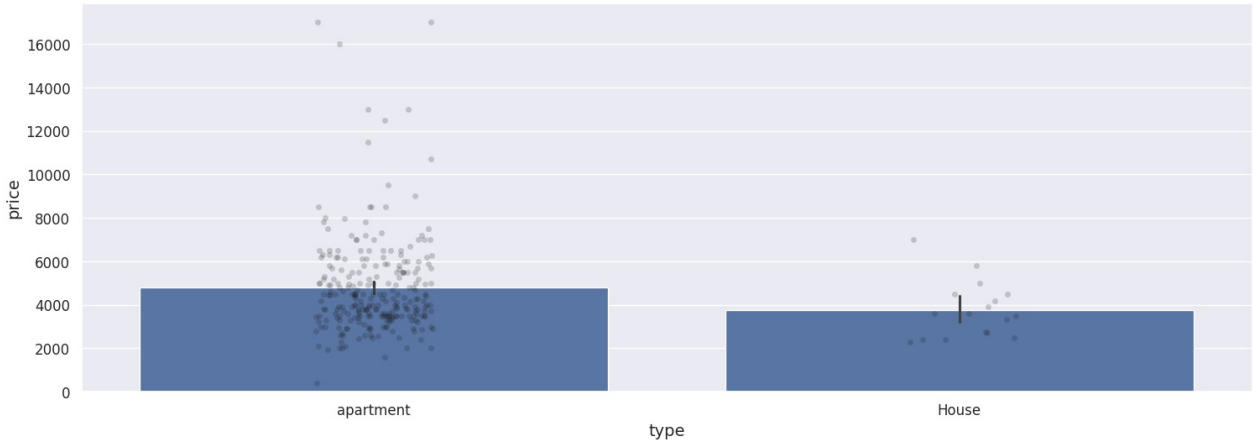
## Question 2

Create a visualization that compares the **average** monthly rates in houses vs. apartments. Which are more expensive on average?

```
plt.figure(figsize=(18,6))
sns.barplot(x='is_a_house', y = "monthlyRate", data=part4_df)
```

```
sns.stripplot(x='is_a_house', y='monthlyRate', data=part4_df, alpha=0.2, color='k')
plt.xlabel("type")
plt.ylabel("price")
ax = plt.gca()
ax.set_xticklabels(['apartment', 'House'])
```

```
<ipython-input-93-0d0aad382fcf>:7: UserWarning: FixedFormatter should only be used
  ax.set_xticklabels(['apartment', 'House'])
[Text(0, 0, 'apartment'), Text(1, 0, 'House')]
```




---

Part 4 Question 2 - textual Answer:

נראה כי בממוצע דירות נוטות להיות ייקרות יותר.

---

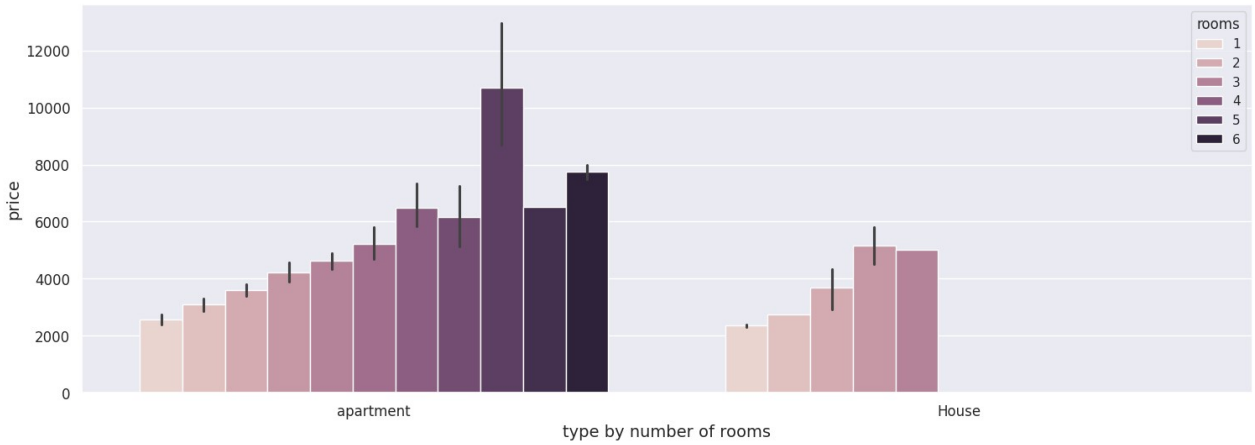
## Question 3

Now, let's look at the data in a higher resolution. Create a visualization that compares the average monthly rates of houses vs. apartments separetly for any number of rooms. Do the results align with the results from the previous question?

results align with the results from the previous question?

```
# Part 4 - Question 3
plt.figure(figsize=(18,6))
sns.barplot(x='is_a_house', y='monthlyRate', hue='rooms', data=part4_df)
plt.xlabel("type by number of rooms")
plt.ylabel("price")
ax = plt.gca()
ax.set_xticklabels(['apartment', 'House'])
```

```
<ipython-input-79-f4a03ff7c08b>:7: UserWarning: FixedFormatter should only be used
  ax.set_xticklabels(['apartment', 'House'])
[Text(0, 0, 'apartment'), Text(1, 0, 'House')]
```




---

## Part 4 Question 3 - textual Answer:

נראה כי בבתים פרטיים יש פחות חדרים ובגלל זה המחירים נמוכים יותר בממוצע.

---

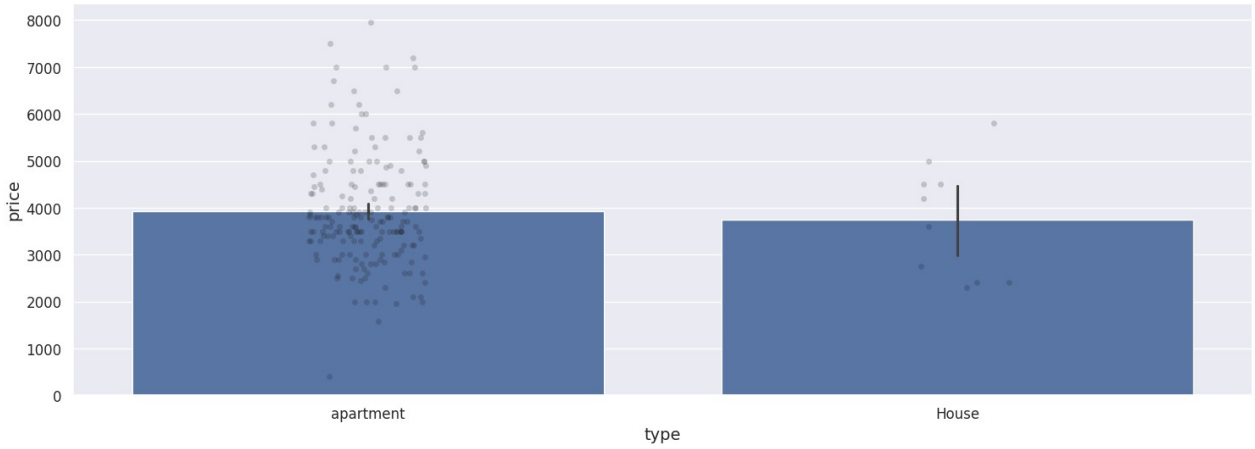
## Question 4

Dan saw those visualizations and suggested that the trend in **question 2** is due to the fact that apartments in this dataset have larger maximal number of rooms than houses.

Create a new visualization similar to **question 2**, but consider only apartment listings with a number of rooms less or equal to the maximal number of rooms for a private house listing. Does the result now align with the trend in **question 3**? If not, is the discrapancy smaller than before?

```
# Part 4 - Question 4
plt.figure(figsize=(18,6))
max_houses = max(part4_df['rooms'][part4_df['is_a_house']== True])
up_to_max = part4_df[part4_df['rooms'] <= max_houses]
sns.barplot(x='is_a_house', y = "monthlyRate", data=up_to_max)
sns.stripplot(x='is_a_house', y='monthlyRate', data=up_to_max, alpha=0.2, color='k')
plt.xlabel("type")
plt.ylabel("price")
ax = plt.gca()
ax.set_xticklabels(['apartment', 'House'])
```

```
<ipython-input-82-1e26e1e7eb25>:10: UserWarning: FixedFormatter should only be use
  ax.set_xticklabels(['apartment', 'House'])
[Text(0, 0, 'apartment'), Text(1, 0, 'House')]
```



---

Part 4 Question 4 - textual Answer:

אחרי שהורדנו בתים עם יותר חדרים מהמקסימום של הבתים הפרטיים נראה כי הממוצעים כן שווים.

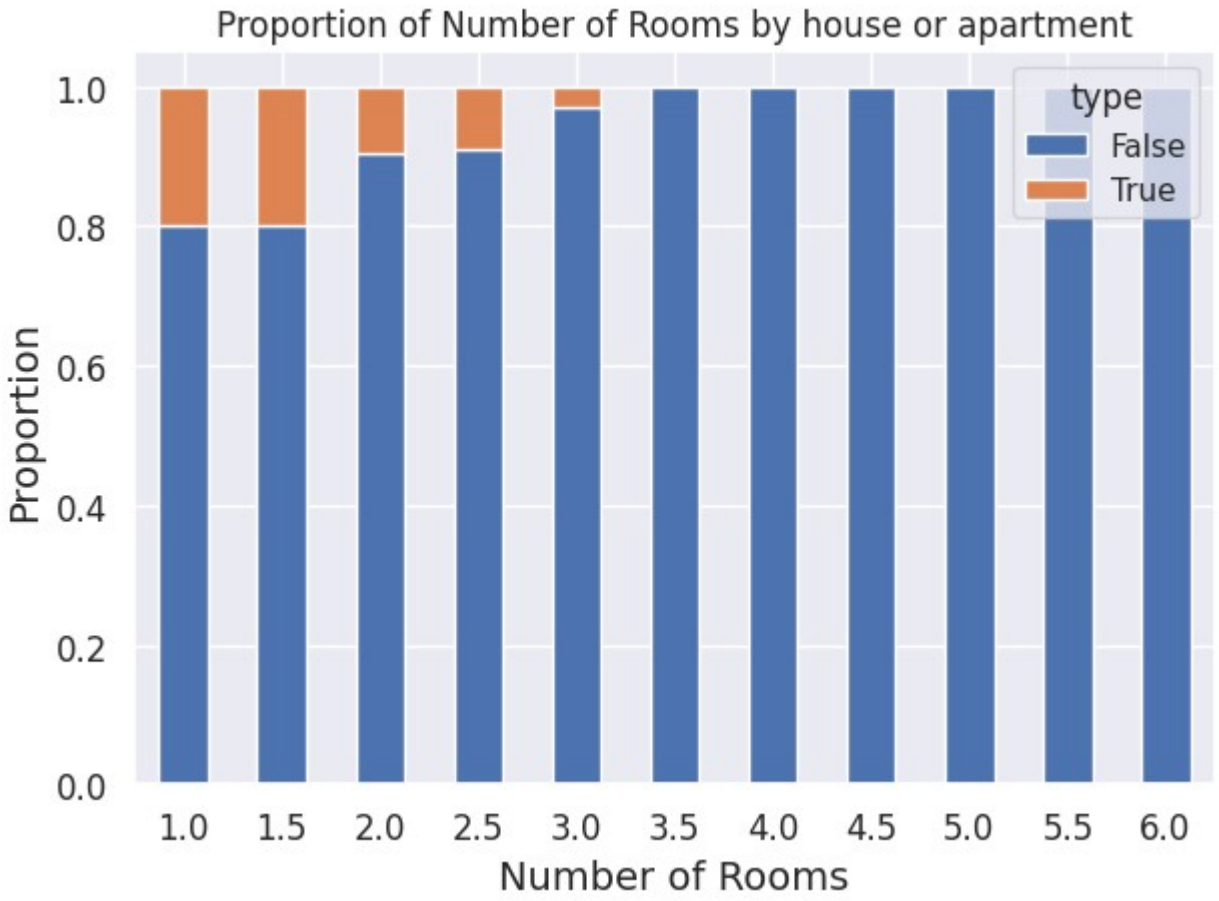
---

## Question 5

Create a visualization that compares the proportion of listings with every value of "number of rooms" in each of the two groups (is_a_house == True and is_a_house == False). How can the results here explain the discrapancy between the results of **question 2** and **question 3**? (Hint: recall the UC Berkeley admission rates example from the first lecture)

```
# Part 4 - Question 5
plt.figure(figsize=(18,6))
prop_table = part4_df.groupby('rooms')['is_a_house'].value_counts(normalize=True).unst
prop_table.plot(kind='bar', stacked=True)
plt.title('Proportion of Number of Rooms by house or apartment')
plt.xlabel('Number of Rooms')
plt.ylabel('Proportion')
plt.xticks(rotation=0)
plt.legend(title='type')
plt.tight_layout()
```

```
<Figure size 1800x600 with 0 Axes>
```



---

Part 4 Question 5 - textual Answer:

הנתונים פה מסבירים את ההבדל בין התשובות השונות שיצאו לנו כי אנחנו צריכים להסתכל על שיעורי הבסיס שלנו ועל כמה תצפיות יש לנו מכל אחד מהסוגים כדי להגיע להחלטה. יש יותר דירות גדולות יותר שיטו לנו את המחירים למעלה והבתים שמושכרים הם יחסית עם מעט חדרים.

---