In [1]:

```
import numpy as np
import pandas as pd
import scipy as sp
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns; sns.set(style='whitegrid')
```

In [2]:

```
url="https://raw.githubusercontent.com/zhukpm/da_spring2021/master/data/lw2/projects
data = pd.read_csv(url)
R1='life_sq'
R2='num_room'
N=45080
t_param='num_room'
t_value=4
perc = 0.973
```

In [3]:

```
data
```

Out[3]:

	full_sq	life_sq	material	num_room
0	42	19	1	1
1	32	18	1	1
2	28	15	1	1
3	59	37	2	2
4	75	44	1	3
...	...	...	...	...
7863	86	50	1	3
7864	39	20	1	1
7865	90	90	6	3
7866	83	83	1	3
7867	36	20	5	1

7868 rows × 4 columns

In [4]:

```
R=data[R1].sum()/data[R2].sum()
R
```

Out[4]:

```
17.98768926925609
```

In [5]:

```
n=len(data)
n
```

Out[5]:

```
7868
```

In [6]:

```
f=n/N
f
```

Out[6]:

```
0.17453416149068324
```

In [7]:

```
y = data[R1].values
x = data[R2].values
```

In [8]:

```
A=(x*x).sum()*R**2+(y*y).sum()-2*R*(x*y).sum()
B=A/(n-1)
C=np.sqrt(B)
S=(np.sqrt(1-f)*C)/(np.sqrt(n)*x.mean())
S
```

Out[8]:

```
0.07590247777682296
```

In [9]:

```
p = data[data[t_param] == t_value].shape[0] / data.shape[0]
q = 1 - p
p
```

Out[9]:

```
0.020589730554143367
```

In [10]:

```
t = sp.stats.norm.ppf((1 + perc) / 2)
left = p - t * np.sqrt(1 - f) * np.sqrt(p * q / (n - 1)) - 1 / (2 * n)
right = p + t * np.sqrt(1 - f) * np.sqrt(p * q / (n - 1)) + 1 / (2 * n)
left, right
```

Out[10]:

```
(0.017309241144118477, 0.023870219964168257)
```

In [11]:

```
sub = {'R' : [R], 'S' : [S], 'p': [p], 'left': [left], 'right': [right]}
df = pd.DataFrame(data=sub)
df.to_csv('answers.csv', index=False)
```

In [12]:

```
df
```

Out[12]:

	R	S	p	left	right
0	17.987689	0.075902	0.02059	0.017309	0.02387

In [ ]: