基于 Airbnb 网站数据的数据可视化

一. 导入依赖以及全局设置

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from PIL import Image
%matplotlib inline
plt.rcParams['font.sans-serif'] = ['SimHei']  # windows用SimHei 用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False  # 用来正常显示负号
plt.rcParams.update({'font.size':13 })
```

```
from pyecharts.globals import CurrentConfig, NotebookType
CurrentConfig.NOTEBOOK_TYPE = NotebookType.JUPYTER_LAB
from pyecharts import options as opts
from pyecharts.charts import Geo
from pyecharts.charts import Line,Scatter
```

二. 读入数据集

```
calendar = pd.read_csv('./data/calendar.csv')
calendar2015 = pd.read_csv('./data/calendar2015.csv')
calendar2016 = pd.read_csv('./data/calendar2016.csv')
calendar2017 = pd.read_csv('./data/calendar2017.csv')
listings = pd.read_csv('./data/listings.csv',low_memory=False)
reviews1 = pd.read_csv('./data/reviews1.csv')
reviews2 = pd.read_csv('./data/reviews2.csv')
sampledreviews = pd.read_csv('./data/sampledreviews.csv')
```

三. 数据预处理

1. listings 数据集(房屋张贴表)

1)查看 listings 数据集信息

```
listings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50968 entries, 0 to 50967
Data columns (total 96 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   id                                50968 non-null  int64
 1   listing_url                       50968 non-null  object
 2   scrape_id                         50968 non-null  int64
 3   last_scraped                      50968 non-null  object
 4   name                              50947 non-null  object
 5   summary                           49038 non-null  object
 6   space                             34971 non-null  object
 7   description                       50500 non-null  object
 8   experiences_offered               50968 non-null  object
 9   neighborhood_overview             30258 non-null  object
 10  notes                             20323 non-null  object
```

```
 11  transit                         31567 non-null  object
 12  access                          29362 non-null  object
 13  interaction                     28255 non-null  object
 14  house_rules                     30358 non-null  object
 15  thumbnail_url                   0 non-null      float64
 16  medium_url                      0 non-null      float64
 17  picture_url                     50968 non-null  object
 18  xl_picture_url                  0 non-null      float64
 19  host_id                         50968 non-null  int64
 20  host_url                        50968 non-null  object
 21  host_name                       50960 non-null  object
 22  host_since                      50960 non-null  object
 23  host_location                   50814 non-null  object
 24  host_about                      31052 non-null  object
 25  host_response_time              26465 non-null  object
 26  host_response_rate              26465 non-null  object
 27  host_acceptance_rate            0 non-null      float64
 28  host_is_superhost               50960 non-null  object
 29  host_thumbnail_url              50960 non-null  object
 30  host_picture_url                50960 non-null  object
 31  host_neighbourhood              43816 non-null  object
 32  host_listings_count             50960 non-null  float64
 33  host_total_listings_count       50960 non-null  float64
 34  host_verifications              50968 non-null  object
 35  host_has_profile_pic            50960 non-null  object
 36  host_identity_verified          50960 non-null  object
 37  street                          50968 non-null  object
 38  neighbourhood                   48407 non-null  object
 39  neighbourhood_cleansed          50968 non-null  object
 40  neighbourhood_group_cleansed    50968 non-null  object
 41  city                            50903 non-null  object
 42  state                           50955 non-null  object
 43  zipcode                         50189 non-null  object
 44  market                          50831 non-null  object
 45  smart_location                  50968 non-null  object

 46  country_code                    50968 non-null  object
 47  country                         50968 non-null  object
 48  latitude                        50968 non-null  float64
 49  longitude                       50968 non-null  float64
 50  is_location_exact               50968 non-null  object
 51  property_type                   50968 non-null  object
 52  room_type                       50968 non-null  object
 53  accommodates                    50968 non-null  int64
 54  bathrooms                       50870 non-null  float64
 55  bedrooms                        50912 non-null  float64
 56  beds                            50907 non-null  float64
 57  bed_type                        50968 non-null  object
 58  amenities                       50968 non-null  object
 59  square_feet                     493 non-null    float64
 60  price                           50968 non-null  object
 61  weekly_price                    7131 non-null   object
 62  monthly_price                   6252 non-null   object
 63  security_deposit                31601 non-null  object
 64  cleaning_fee                    38762 non-null  object
 65  guests_included                 50968 non-null  int64
 66  extra_people                    50968 non-null  object
 67  minimum_nights                  50968 non-null  int64
 68  maximum_nights                  50968 non-null  int64
 69  calendar_updated                50968 non-null  object
 70  has_availability                50968 non-null  object
 71  availability_30                 50968 non-null  int64
 72  availability_60                 50968 non-null  int64
 73  availability_90                 50968 non-null  int64
 74  availability_365                50968 non-null  int64
 75  calendar_last_scraped           50968 non-null  object
```

```
 76  number_of_reviews                 50968 non-null  int64
 77  first_review                      40341 non-null  object
 78  last_review                       40343 non-null  object
 79  review_scores_rating              39254 non-null  float64
 80  review_scores_accuracy            39205 non-null  float64
 81  review_scores_cleanliness         39226 non-null  float64
 82  review_scores_checkin             39178 non-null  float64
 83  review_scores_communication       39211 non-null  float64
 84  review_scores_location            39171 non-null  float64
 85  review_scores_value               39172 non-null  float64
 86  requires_license                  50968 non-null  object
 87  license                           6 non-null      object
 88  jurisdiction_names                30 non-null     object
 89  instant_bookable                  50968 non-null  object
 90  is_business_travel_ready          50968 non-null  object
 91  cancellation_policy               50968 non-null  object
 92  require_guest_profile_picture     50968 non-null  object
 93  require_guest_phone_verification  50968 non-null  object
 94  calculated_host_listings_count    50968 non-null  int64
 95  reviews_per_month                 40341 non-null  float64
dtypes: float64(20), int64(13), object(63)
memory usage: 37.3+ MB
```

2) 从 listings 数据集中取出需要进行数据可视化分析的列，存入 listings_中并且重命名所选列

```
# 从listings数据集中取出需要进行数据可视化分析的列，存入listings_中
listings_ = listings.loc[:,['id','host_id','host_name','host_since','host_is_superhost','host_listings_count',
                            'neighbourhood_cleansed','neighbourhood_group_cleansed','state','availability_365',
                            'property_type','room_type','bedrooms','bed_type','price',
                            'security_deposit','cleaning_fee','number_of_reviews','review_scores_rating']].copy()
```

```
# 重命名listings_的列名，改成对应的中文命名
listings_.rename(columns={'host_id':'房屋ID','host_name':'房屋名称','host_since':'房屋建立时间','host_is_superhost':'是否为大房屋',
                          'host_listings_count':'房屋租赁数量','neighbourhood_cleansed':'地址','neighbourhood_group_cleansed':'地区',
                          'state':'州','availability_365':'可租总天数/年','property_type':'房屋类型','room_type':'房间类型',
                          'bedrooms':'卧室数量','bed_type':'床类型','price':'价格','security_deposit':'保证金',
                          'cleaning_fee':'清理费用','number_of_reviews':'评论数量','review_scores_rating':'评分'},inplace=True)
```

```
listings_.sample(5)
```

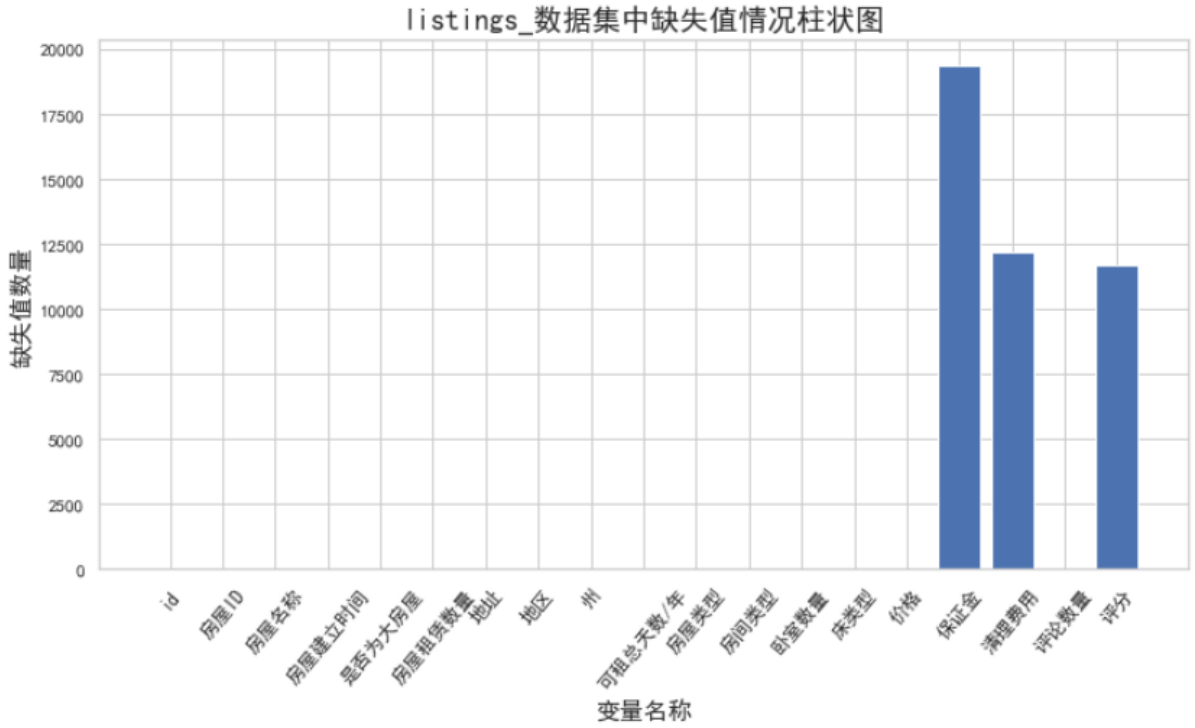
	id	房屋ID	房屋名称	房屋建立时间	是否为大房屋	房屋租赁数量	地址	地区	州	可租总天数/年	房屋类型	房间类型	卧室数量	床类型	价格	保证金	清理费用	评论数量	评分	建房年数
1264	416426	970385	Ted	2011-08-14	t	3.0	UpperWestSide	Manhattan	New York	35	Apartment	Entirehome/apt	2	RealBed	349.0	500.0	100.0	17	92.0	9
30108	19346710	135559842	Mauricio	2017-06-17	f	1.0	Harlem	Manhattan	New York	0	Apartment	Privateroom	1	RealBed	49.0	0.0	0.0	0	0.0	3
6494	3946288	20438402	Geoff&Aurelia	2014-08-23	f	3.0	EastHarlem	Manhattan	New York	351	Apartment	Entirehome/apt	3	RealBed	529.0	600.0	200.0	19	91.0	6
8468	5106041	4265630	NaomiAndRashid	2012-11-27	t	2.0	Flatbush	Brooklyn	New York	338	Apartment	Entirehome/apt	1	RealBed	80.0	200.0	85.0	37	92.0	8
38493	23018551	101051232	Jr	2016-10-24	f	1.0	Prospect-LeffertsGardens	Brooklyn	New York	144	Loft	Entirehome/apt	1	RealBed	80.0	0.0	50.0	17	81.0	4

3) 查看空值情况

```
listings_.isnull().sum()
```

```
id              0
房屋ID            0
房屋名称            8
房屋建立时间          8
是否为大房屋          8
房屋租赁数量          8
地址              0
地区              0
州              13
可租总天数/年         0
房屋类型            0
房间类型            0
卧室数量           56
床类型             0
价格              0
保证金          19367
清理费用         12206
评论数量            0
评分           11714
dtype: int64
```

```
plt.figure(figsize=(12,6))
plt.bar(listings_.columns,listings_.isnull().sum())
plt.xticks(fontsize=12,rotation=50)
plt.ylabel('缺失值数量',fontsize=15)
plt.xlabel('变量名称',fontsize=15)
plt.title('listings_数据集中缺失值情况柱状图',fontsize=18)
plt.show()
```



4) 数据清洗

```
# 删除房屋位置信息缺少的数据
listings_.dropna(subset=['房屋名称','地区','州'],how='any',inplace=True)
```

```
# 对卧室数量缺少的数据填充 0
listings_.卧室数量.fillna(0,inplace=True)
```

去除多余的占位符

```
listings_.loc[:,['价格','保证金','清理费用']] = listings_.loc[:,['价格','保证金','清理费用']].replace('\t','')
listings_.loc[:,['价格','保证金','清理费用']] = listings_.loc[:,['价格','保证金','清理费用']].replace('\n','')
```

将空值替换成'$0.00'

```
listings_.loc[:,['价格','保证金','清理费用']] = listings_.loc[:,['价格','保证金','清理费用']].replace(np.NaN,'$0.00')
```

去除美元符号

```
listings_['价格'] = listings_['价格'].map(lambda x:str(x)[1:])
listings_['保证金'] = listings_['保证金'].map(lambda x:str(x)[1:])
listings_['清理费用'] = listings_['清理费用'].map(lambda x:str(x)[1:])
```

转换类型

```
# 去除价格的会计格式，例如'1,000.00',将其转成 1000.00
listings_.loc[:,['价格','保证金','清理费用']] = listings_.loc[:,['价格','保证金','清理费用']].applymap(lambda x:x.replace(',',''))
```

```
listings_['价格'] = listings_['价格'].astype('float64')
listings_['保证金'] = listings_['保证金'].astype('float64')
listings_['清理费用'] = listings_['清理费用'].astype('float64')
```

处理没有评论的数据，将评分设为0分

```
listings_.评分.fillna(0,inplace=True)
```

去除空格符

```
object_lt = listings_.dtypes[listings_.dtypes=='object'].index.to_list()

listings_.loc[:,object_lt] = listings_.loc[:,object_lt].applymap(lambda x:x.replace(' ',''))
```

对州数据进行清洗汇总

将'NY','ny','NewYork','Ny'替换成'New York'
将'MP'替换成'Northern Mariana Islands'
将'CA'替换成'California'
将'NJ'提换成'New Jersey'

```
# 清洗前
listings_.州.unique()
```

```
array(['NY', 'ny', 'MP', 'NewYork', 'CA', 'NJ', 'Ny'], dtype=object)
```

```
# 对州数据进行清洗
for i in range(len(listings_)):
    temp = listings_.iloc[i,8]
    if temp in ['NY','ny','NewYork','Ny']:
        listings_.iloc[i,8] = 'New York'
    if temp == 'CA':
        listings_.iloc[i,8] = 'California'
    if temp == 'NJ':
        listings_.iloc[i,8] = 'New Jersey'
    if temp == 'MP':
        listings_.iloc[i,8] = 'Northern Mariana Islands'
```

```
# 清洗后
listings_.州.unique()
```

```
array(['New York', 'Northern Mariana Islands', 'California', 'New Jersey'],
      dtype=object)
```

城市数据

纽约五大区：Queens,Brooklyn,Staten Island,Manhattan,The Bronx

```
listings_.地区.unique()
```

```
array(['Brooklyn', 'Manhattan', 'Queens', 'Bronx', 'StatenIsland'],
      dtype=object)
```

建房年数

采用2020年为基准年，构造建房年数，用于表示房屋建成至2020的年数

```
listings_['建房年数'] = listings_.房屋建立时间.apply(lambda x:(2020 - eval(x.split('-')[0])))
```

```
# 转换类型
listings_.卧室数量 = listings_.卧室数量.astype('int')
```

```
listings_.sample(5)
```

	id	房屋ID	房屋名称	房屋建立时间	是否为大房屋	房屋租赁数量	地址	地区	州	可租总天数/年	房屋类型	房间类型	卧室数量	床类型	价格	保证金	清理费用	评论数量	评分	建房年数
30113	19332077	4298040	Jon	2012-12-01	f	1.0	FortGreene	Brooklyn	New York	0	Apartment	Privateroom	1	RealBed	125.0	0.0	0.0	0	0.0	8
32006	20257385	144036514	Chris	2017-08-03	f	5.0	Bushwick	Brooklyn	New York	0	Townhouse	Privateroom	1	RealBed	45.0	200.0	0.0	1	100.0	3
34581	21487004	87236554	Annie	2016-08-01	t	1.0	EastVillage	Manhattan	New York	0	Apartment	Privateroom	1	RealBed	103.0	0.0	30.0	39	96.0	4
15540	9793465	13811334	Rachel	2014-04-02	f	1.0	EastHarlem	Manhattan	New York	0	Apartment	Entirehome/apt	1	RealBed	140.0	150.0	100.0	1	100.0	6
15374	9753165	2436118	Bianca	2012-05-22	f	1.0	Bushwick	Brooklyn	New York	0	Apartment	Privateroom	1	RealBed	60.0	100.0	50.0	0	0.0	8

数据清洗后查看数据集listings_的清洗情况

```
listings_.isnull().sum().sum()
```

```
0
```

2. reviews(评论表)

reviews(评论表)

拼接reviews1和reviews2

```
reviews = pd.concat([reviews1,reviews2],axis=0)
reviews
```

	listing_id	id	date	reviewer_id	reviewer_name	comments
0	2515	198	2008-10-13	2603	Jenny	Stephanie was a wonderful host! Her apartment ...
1	2515	859	2009-03-08	8455	Roland	Such a wonderful place and very close to the m...
2	2515	1083	2009-03-25	9759	Cem	I just got back from a trip to NYC during whic...
3	2515	1107	2009-03-27	9193	Holly	Stephanie's offered all the most important thi...
4	2515	2175	2009-05-09	7048	Alessandra	Stephanie was really nice, ftiendly and helpfu...
...	...	...	...	...	...	...
495496	6292029	229000816	2018-01-21	32877224	Andrew	The space is really inspired by/for artists an...
495497	6292029	310490619	2018-08-19	135171825	Barbara	You will find splendid people, good vibrations...
495498	24130099	338102424	2018-10-18	49095948	Jerry	Fantastic location at CPS/Columbus Circle. Goo...
495499	710040	2501059	2012-10-03	3377326	Romain & Roxane	The price was to high for the confort, but Ash...
495500	710040	2552043	2012-10-08	3385316	Yenia.Yeya	She was very kind and nice. my friend and I fe...

1094718 rows × 6 columns

3. calendar(预约表)

1)合并数据集

```
# 合并数据
calendar_ = pd.concat([calendar2015,calendar2016,calendar2017,calendar],axis=0,ignore_index=True)
calendar_.sample(5)
```

	listing_id	date	available	price
34052153	12628500	2018-03-08	f	NaN
4998241	8880173	2016-08-23	t	$225.00
25805392	13857521	2017-02-02	f	NaN
33217797	10877492	2018-09-30	f	NaN
37148420	16394700	2018-01-22	f	NaN

2) 数据清洗

```
# 备份数据
calendar_df = calendar_.copy()
```

```
# 删除无价格的无效数据
calendar_df.dropna(subset='price',inplace=True)
```

```
# 转换时间序列
calendar_df.date = pd.to_datetime(calendar_df.date)
calendar_df['year'] = calendar_df.date.dt.year
calendar_df['month'] = calendar_df.date.dt.month
calendar_df.head()
```

	listing_id	date	available	price	year	month
29	7617325	2015-11-30	t	$50.00	2015	11
30	7617325	2015-12-01	t	$50.00	2015	12
31	7617325	2015-12-02	t	$50.00	2015	12
32	7617325	2015-12-03	t	$50.00	2015	12
33	7617325	2015-12-04	t	$50.00	2015	12

```
# 清理价格数据，取出'$'符号
calendar_df.price = calendar_df.price.apply(lambda x:eval(x.split('$')[1]))
calendar_df.sample(5)
```

	listing_id	date	available	price	year	month
21043550	8645816	2017-03-16	t	46.0	2017	3
47684331	9341085	2019-04-26	t	95.0	2019	4
22257022	8499719	2017-09-10	t	70.0	2017	9
15168671	1396458	2017-10-31	t	199.0	2017	10
29914363	6105742	2018-06-29	t	150.0	2018	6

```
# 清理价格是价格范围的情况，如(1,100)
def clear_price(p):
    if ',' in p :
        min_ = eval(p.split(',')[0].split('(')[1])
        max_ = eval(p.split(',')[1].split(')')[0])
        return (min_+max_)/2
    else:
        return p
```

```
# 应用 clear_price 函数
calendar_df.price = calendar_df.price.astype(str).apply(clear_price)
```

```
# 将价格转换类型为float
calendar_df.price = calendar_df.price.astype('float64')
```

```
# 查看清理情况
calendar_df.isnull().sum()
```

```
listing_id    0
date          0
available     0
price         0
year          0
month         0
dtype: int64
```

四. 数据分析以及可视化

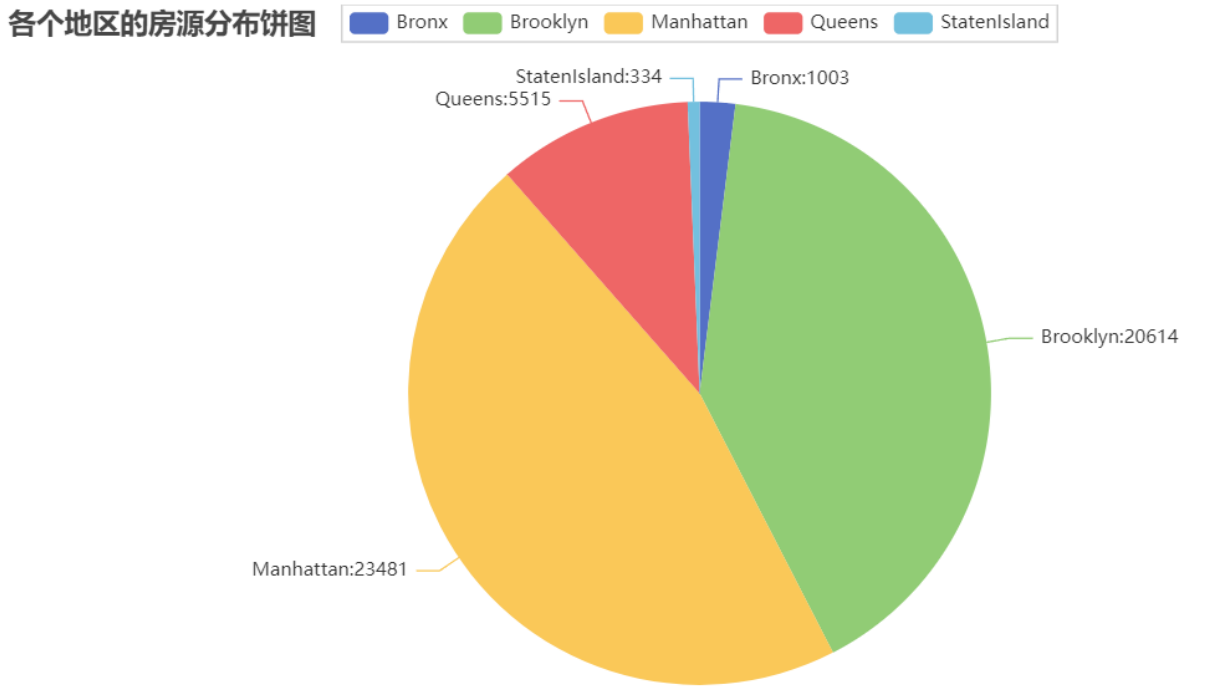
1. 查看纽约各个地区的房源数量，并且绘制饼图

```
data1 = listings_.groupby('地区')[['房屋ID']].count()
display(data1)
```

地区	房屋ID
Bronx	1003
Brooklyn	20614
Manhattan	23481
Queens	5515
StatenIsland	334

```
from pyecharts import options as opts
from pyecharts.charts import Pie
#画图
pie = Pie()
pie.add('地区分布饼图',
        [list(z) for z in zip(data1.index.tolist(), data1.values[:,0].tolist())],
        label_opts=opts.LabelOpts(is_show=True)
       )
pie.set_global_opts(title_opts=opts.TitleOpts(title="各个地区的房源分布饼图"))
pie.set_series_opts(label_opts=opts.LabelOpts(formatter="{b}:{c}"))
pie.load_javascript()
```

```
#渲染
pie.render_notebook()
```



分析：从上述饼图可以清晰的看出，纽约各个地区的房源数量分布排名情况是：Manhattan，Brooklyn，Queens，Bronx，StatenIsland

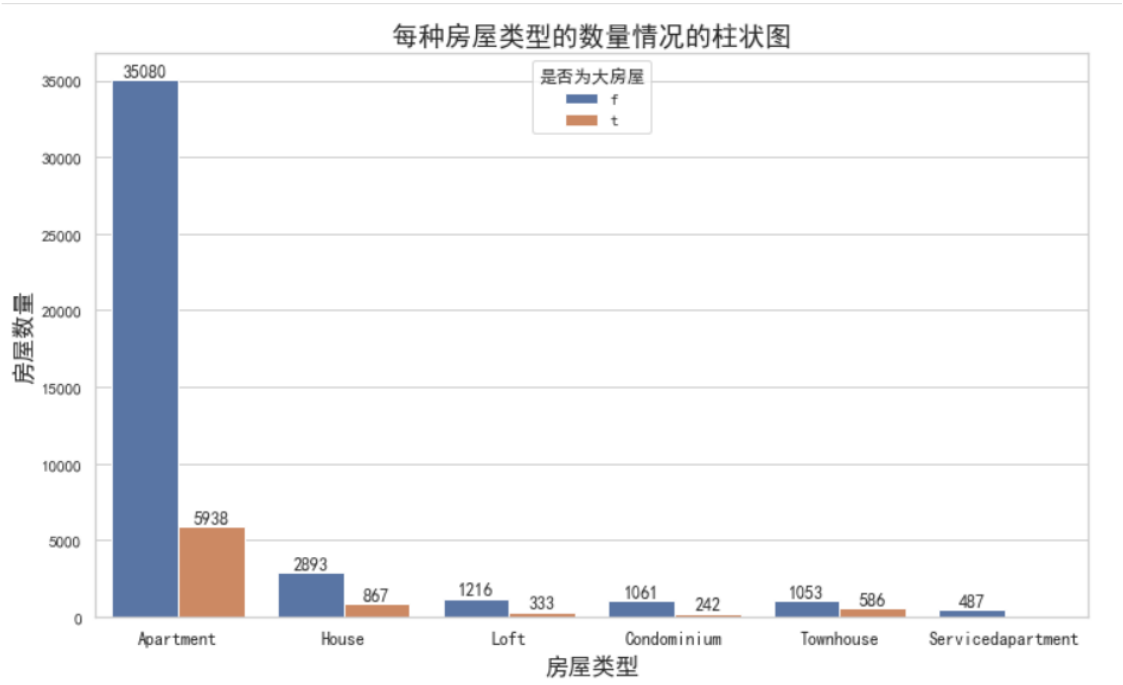
2. 查看每种房屋类型的数量情况，要求降序输出并且绘制排序前十的柱状图

```
data2 = listings_.groupby(['房屋类型','是否为大房屋'])[['房屋ID']].count()
data2.sort_values(by='房屋ID',ascending=False,inplace=True)
```

```
# 取出前十个
data2 = data2[0:11]
# 重设索引
data2 = data2.reset_index()
```

```
data2
```

	房屋类型	是否为大房屋	房屋ID
0	Apartment	f	35080
1	Apartment	t	5938
2	House	f	2893
3	Loft	f	1216
4	Condominium	f	1061
5	Townhouse	f	1053
6	House	t	867
7	Townhouse	t	586
8	Servicedapartment	f	487
9	Loft	t	333
10	Condominium	t	242



分析：房屋类型中 Apartment(公寓)所占的数量最多，其数量远超过其他类型的房屋数量；同时从上述柱状图也可看出大房屋的占比远不如小房屋。

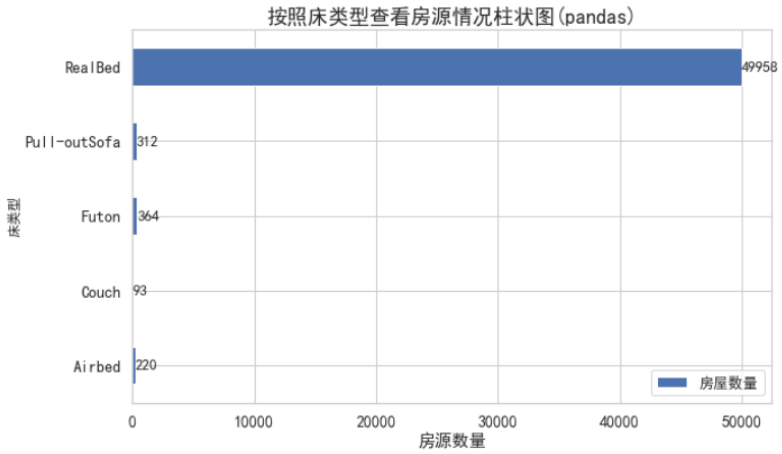
3. 利用数据透视表，查看不同房间类型和床类型的房源数量情况

```
pd.pivot_table(listings_,index=['房间类型','床类型'],values='房屋ID',aggfunc='count').rename(columns={'房屋ID':'房屋数量'}).T
```

房间类型	Entirehome/apt					Privateroom					Sharedroom				
床类型	Airbed	Couch	Futon	Pull-outSofa	RealBed	Airbed	Couch	Futon	Pull-outSofa	RealBed	Airbed	Couch	Futon	Pull-outSofa	RealBed
房屋数量	61	19	107	112	26280	131	19	217	130	22714	28	55	40	70	964

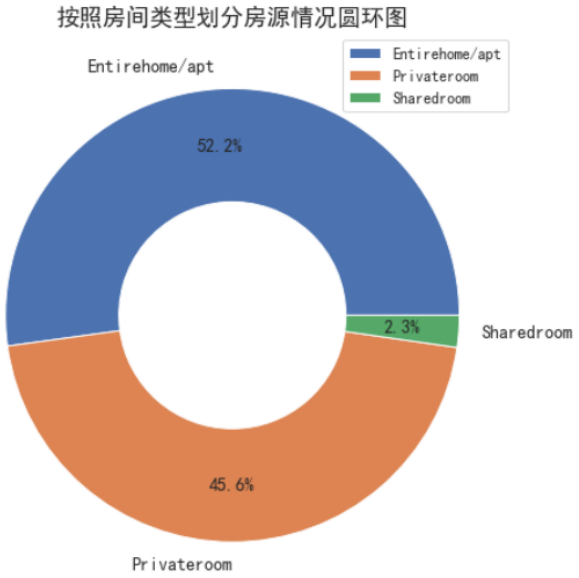
按照床类型查看房源情况柱状图(pandas 绘制)

```
a=listings_.groupby('床类型')[['房屋ID']].count().rename(columns={'房屋ID':'房屋数量'}).plot(kind='barh',
                                                        figsize=(10,6),fontsize=14)
plt.bar_label(a.containers[0])
plt.legend(loc="lower right",fontsize=13)
plt.title('按照床类型查看房源情况柱状图(pandas)',fontsize=18)
plt.xlabel('房源数量',fontsize=15)
plt.show()
```



按照房间类型查看房源情况圆环图：

```
plt.figure(figsize=(10,8))
data = listings_.groupby('房间类型')[['房屋ID']].count()
pie_labels = data.index
plt.pie(data.房屋ID,radius=1,wedgeprops={'width':0.5},labels=pie_labels,
        autopct='%3.1f%%',pctdistance=0.75,textprops={'fontsize':14})
plt.title('按照房间类型划分房源情况圆环图',fontsize=18)
plt.legend(labels=pie_labels,fontsize=12)
plt.show()
```



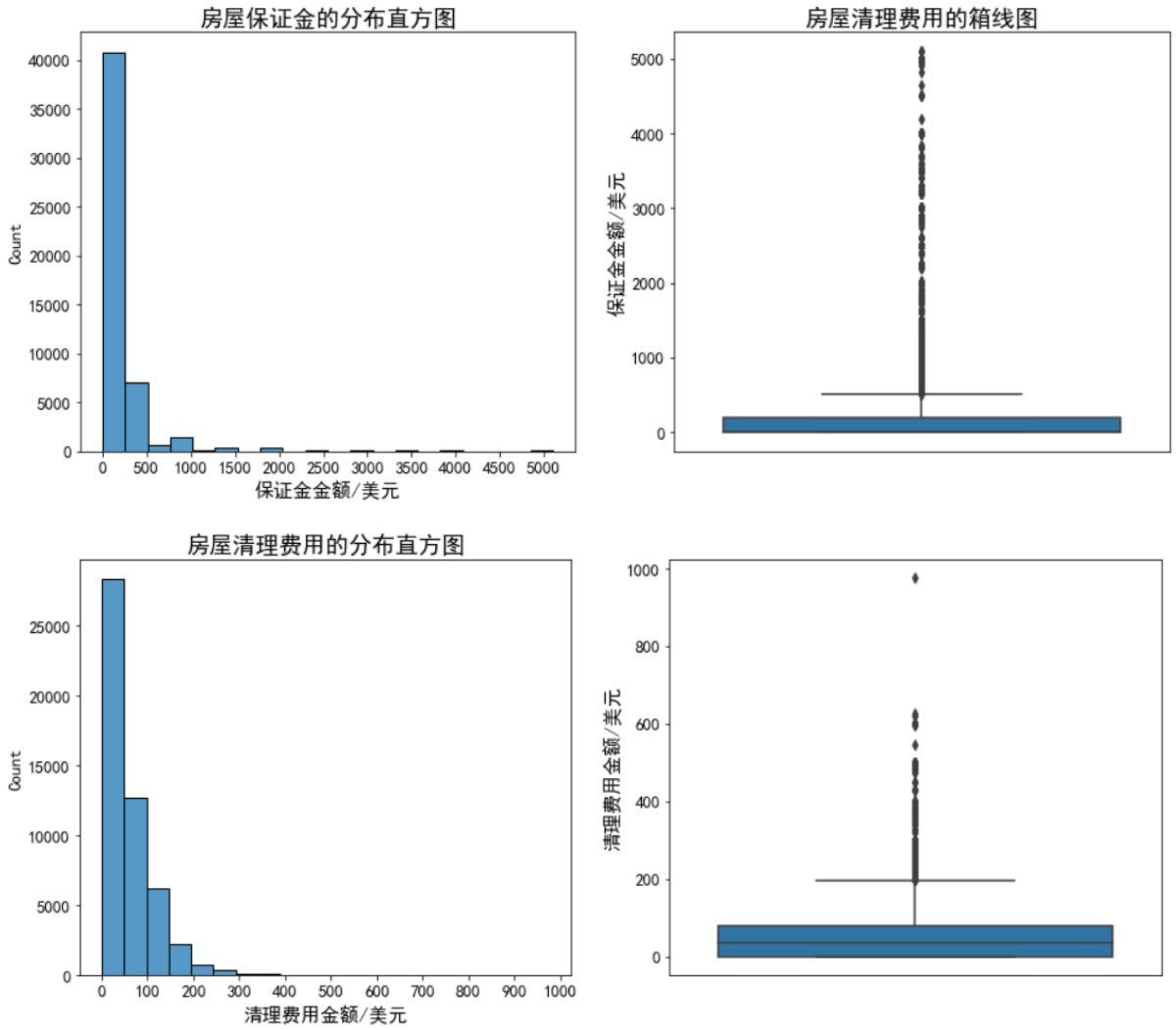
分析：从房间类型看，Entirehome/apt 和 Privateroom 的房源数量占比最多；
从床类型看，RealBed 占比最多，Futon 和 Pull-outSofa 次之，Airbed 和 Couch 占比最少，但是除 RealBed 外的床类型的房屋数量与 RealBed 床类型的房源数量差距悬殊。

4. 求需要保证金和清理费用的房源数量，同时对保证金和清理费用查看其价格分布区间

```
print(f'需要保证金的房屋数量为:{np.count_nonzero(listings_.保证金)};\
占比为:{(round(np.count_nonzero(listings_.保证金)/listings_.房屋ID.count(),4))*100}%')

print(f'需要清理费用的房屋数量为:{np.count_nonzero(listings_.清理费用)};\
占比为:{(round(np.count_nonzero(listings_.清理费用)/listings_.房屋ID.count(),4))*100}%')
```

```
需要保证金的房屋数量为:21366;占比为:41.94%
需要清理费用的房屋数量为:36734;占比为:72.1%
```

```
# 保证金和清理费用的分布情况绘图
plt.figure(figsize=(15,13))

ax1 = plt.subplot(2,2,1)
sns.histplot(listings_.保证金,bins=20)
ax1.set_xlabel('保证金金额/美元',fontsize=15)
ax1.set_xticks([x for x in range(0,5500,500)])
ax1.set_title('房屋保证金的分布直方图',fontsize=18)

ax2 = plt.subplot(2,2,2)
sns.boxplot(listings_.保证金)
ax2.set_xticks([1],labels=['保证金'])
ax2.set_ylabel('保证金金额/美元',fontsize=15)
ax2.set_title('房屋保证金的箱线图',fontsize=18)

ax3 = plt.subplot(2,2,3)
sns.histplot(listings_.清理费用,bins=20)
ax3.set_xlabel('清理费用金额/美元',fontsize=15)
ax3.set_xticks([x for x in range(0,1100,100)])
ax3.set_title('房屋清理费用的分布直方图',fontsize=18)

ax4 = plt.subplot(2,2,4)
sns.boxplot(listings_.清理费用)
ax4.set_xticks([1],labels=['清理费用'])
ax4.set_ylabel('清理费用金额/美元',fontsize=15)
ax2.set_title('房屋清理费用的箱线图',fontsize=18)

plt.show()
```



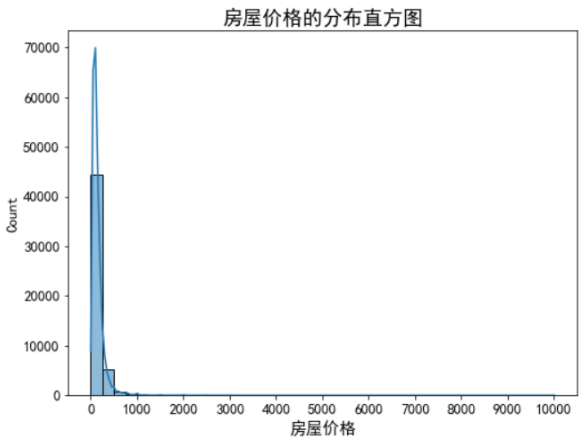
分析：

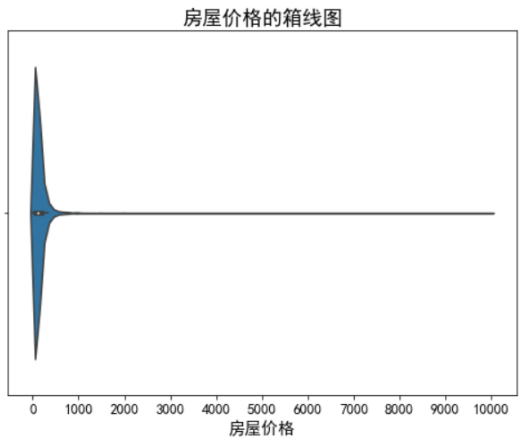
保证金：金额大致集中在 0-1000$之间，但从箱线图看出房源中存在需要大额保证金的部分房屋，最高有需要 5000$的房屋；

清理费用：金额大致集中在 0-200$之间，同样从箱线图看出存在需要较大额清理费用的房屋，最高清理费用可达 1000$左右。

5. 查看房屋价格的分布情况（seaborn 实现）

```
plt.figure(figsize=(18,6))
ax1 = plt.subplot(1,2,1)
ax1 = sns.histplot(data=listings_,x='价格',bins=40,kde=True)
ax1.set_xticks([x for x in range(0,11000,1000)])
ax1.set_xlabel('房屋价格',fontsize=15)
ax1.set_title('房屋价格的分布直方图',fontsize=18)


ax2 = plt.subplot(1,2,2)
sns.violinplot(x=listings_.价格)
ax2.set_xticks([x for x in range(0,11000,1000)])
ax2.set_xlabel('房屋价格',fontsize=15)
ax2.set_title('房屋价格的箱线图',fontsize=18)
plt.show()
```





分析：房屋价格主要分布在 0-1000$之间，但从上述两图形均可看出，房屋价格的总体分布是 0-10000$,不过 1000—10000$的房源数量占比极小。

6. 按照地区将房屋分组，计算平均价格,绘制地区折线图（pyecharts 实现）

```
data = listings_.groupby('地区')[['价格']].mean()
data = data.reset_index()
data['价格'] = data['价格']
```

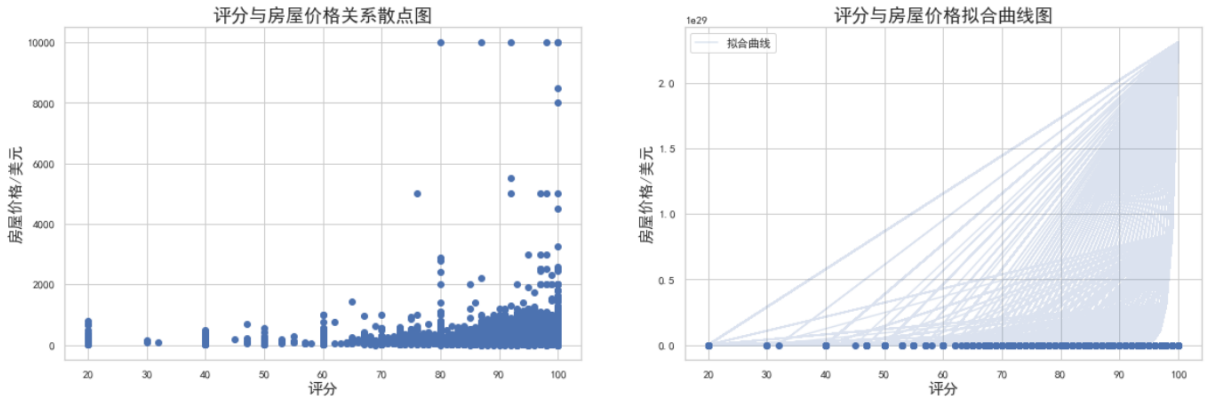
```
data['价格'] = data['价格'].apply(lambda x:round(x,2))
```

```
from pyecharts.charts import Line
line = (Line()
        .add_xaxis(xaxis_data=data['地区'].astype("str").to_list())
        .add_yaxis("房屋平均价格",data['价格'].tolist(),
                   markpoint_opts=opts.MarkPointOpts(
                    data=[opts.MarkPointItem(type_="max", name="最大值"),
                          opts.MarkPointItem(type_="min", name="最小值"),]))
        .set_global_opts(title_opts=opts.TitleOpts(title="各地区房屋平均价格折线图",pos_top="1%"),
                         toolbox_opts = opts.ToolboxOpts(is_show = True),
                         xaxis_opts = opts.AxisOpts(type_='category',name='地区',is_show=True),
                         yaxis_opts = opts.AxisOpts(type_='value',name='房屋平均价格',is_show=True)))
line.load_javascript()
```

分析：从上述折线图可以看出纽约市各地区的房屋平均价格排名情况依次是：Manhattan,StatenIsland,Brooklyn，Queens，Bronx

7. 分析房屋评分和价格的相关性情况，并且利用 scipy 进行相关性拟合

```
from scipy.optimize import curve_fit
plt.figure(figsize=(20,6))
# 取出评分不为0 的有效数据
data = listings_[listings_.评分!=0][['评分','价格']]
X = data.评分
Y = data.价格
# 散点图
ax1 = plt.subplot(1,2,1)
ax1.scatter(X,Y)
ax1.set_xlabel('评分',fontsize=15)
ax1.set_ylabel('房屋价格/美元',fontsize=15)
ax1.set_title('评分与房屋价格关系散点图',fontsize=18)
# 拟合曲线
ax2 = plt.subplot(1,2,2)
# 定义拟合函数
def func(x, a, b, c):
    return a * np.exp(b * x) + c
# 初始参数值
initial_guess = [1, 1, 1]
# 最小二乘法进行曲线拟合
popt, _ = curve_fit(func, X, Y, initial_guess)
ax2.scatter(X,Y)
ax2.plot(X, func(X, *popt), 'b-',alpha=0.2, label='拟合曲线')
ax2.set_xlabel('评分',fontsize=15)
ax2.set_ylabel('房屋价格/美元',fontsize=15)
ax2.set_title('评分与房屋价格拟合曲线图',fontsize=18)
plt.legend()
plt.show()
```



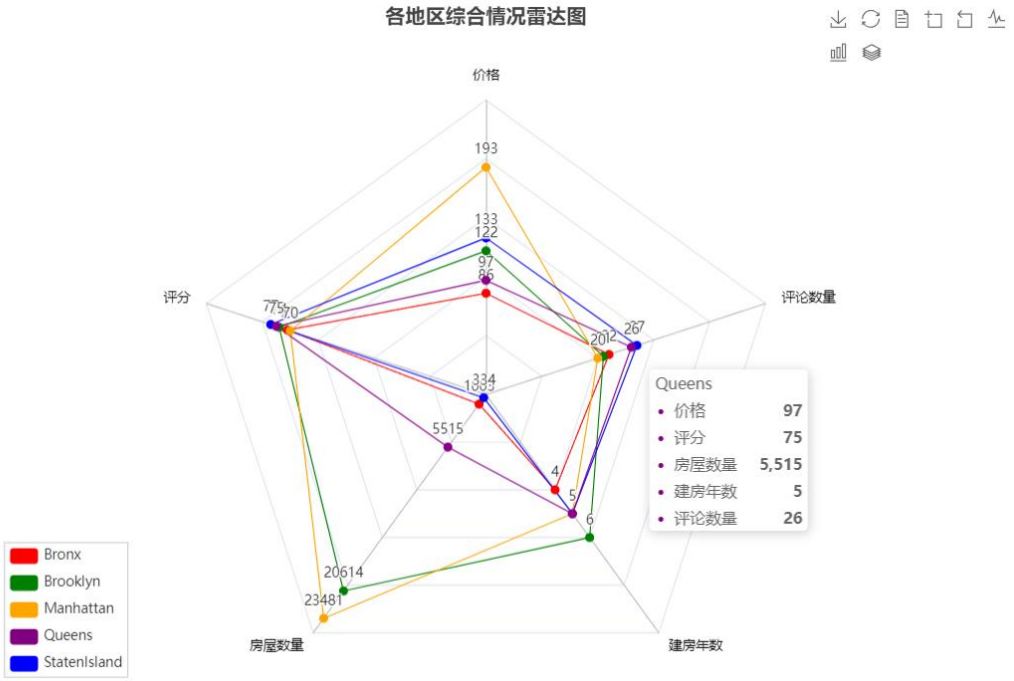
分析:从上述图表亦可看出房屋价格大主要分布区间是 0-1000$,评分的分布区间主要分布在 60-100 之间，从 scipy 的曲线拟合情况来看，房屋价格和评分的相关性大致呈现指数递增趋势，但并不是指数爆炸型增长，增长趋势较为平缓。

8. 各个地区的综合情况评估，并且绘制雷达图

```
# 获取数据
df = listings_.groupby('地区').agg({'价格':'mean','房屋ID':'count','评分':'mean','建房年数':'mean','评论数量':'mean'})
df.rename(columns={'房屋ID':'房屋数量'},inplace=True)
df = df.loc[:,['价格','评分','房屋数量','建房年数','评论数量']].applymap(lambda x:round(x))
df
```

地区	价格	评分	房屋数量	建房年数	评论数量
Bronx	86	71	1003	4	22
Brooklyn	122	74	20614	6	21
Manhattan	193	70	23481	5	20
Queens	97	75	5515	5	26
StatenIsland	133	77	334	5	27

```
import pyecharts.options as op
from pyecharts.charts import Radar

data = df.values
x_schema = [
        {"name": "价格",  "color": 'black', "font_size": 20},
        {"name": "评分",  "color": 'black', "font_size": 20},
        {"name": "房屋数量", "color": 'black', "font_size": 20},
        {"name": "建房年数",  "color": 'black', "font_size": 20},
        {"name": "评论数量",  "color": 'black', "font_size": 20}
    ]
radar_x = Radar(init_opts=opts.InitOpts(width="1000px",height='700px'))
radar_x.add_schema(x_schema)
radar_x.add('Bronx', [data[0].tolist()], color='red').set_colors(['red'])
radar_x.add('Brooklyn', [data[1].tolist()], color='green').set_colors(['green'])
radar_x.add('Manhattan', [data[2].tolist()], color='orange').set_colors(['orange'])
radar_x.add('Queens', [data[3].tolist()], color='purple').set_colors(['purple'])
radar_x.add('StatenIsland', [data[4].tolist()], color='blue').set_colors(['blue'])

radar_x.set_global_opts(
        title_opts=op.TitleOpts(title="各地区综合情况雷达图", pos_right="center"),
        toolbox_opts = opts.ToolboxOpts(is_show = True),
        legend_opts=op.LegendOpts(legend_icon="roundRect", align="left", pos_left='7%',
                                  pos_bottom='14%', orient='vertical')
    )
radar_x.load_javascript()
```



分析：

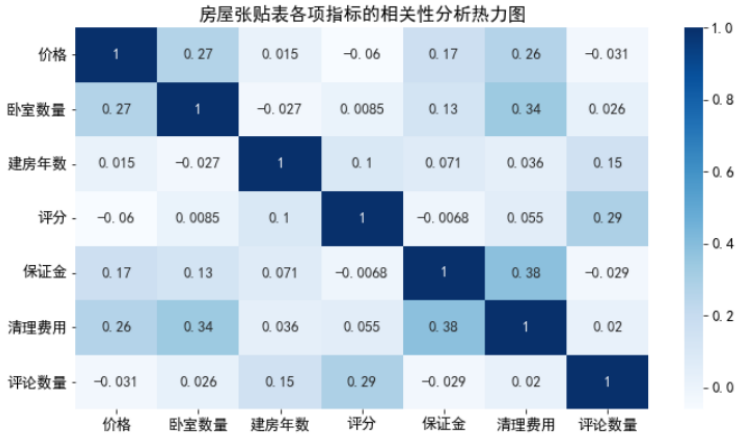
通过绘制雷达图，可以从房屋数量，建房年数，评论数据，评分，房屋价格五个维度评估纽约市五个地区的房源情况；从上述图表得出各地区的房屋情况评估排名是：

Manhattan, Brooklyn, Queens, Bronx,StatenIsland,但 Bronx 和 StatenIsland 的差距并不明显。

9. 各项指标的相关性分析热力图

```
corr_data = listings_.loc[:,['价格','卧室数量','建房年数','评分','保证金','清理费用','评论数量']].corr()
```

```
plt.figure(figsize=(11,6))
sns.heatmap(corr_data,annot=True,cmap=plt.cm.Blues)
plt.title('房屋张贴表各项指标的相关性分析热力图')
plt.show()
```



分析：从上述房屋各项指标的热力图可以清楚看出各项之间的相关性，比如与价格相关性较为明显的是卧室数量，清理费用和保证金；和清理费用相关性较为明显的是卧室数量，保证金和价格。

10. 对于 reviews 的 2018 年的数据进行词云图绘制(评论表的词云图)

由于数据量巨大，因而选取 2018 年的评论数据进行绘制词云图

清洗过程：

```
# 获取2018年评论数据并进行初步清洗
df = reviews.loc[['2018' in x for x in reviews.date],:].copy()
df.dropna(how='any',inplace=True)
df = df.loc[[x[0].isalpha() for x in df.comments],:]
df.comments = df.comments.apply(lambda x:x.replace('.','').replace('/','').replace('\n','').
                                replace(',','').replace('(','').replace(')','').replace('-','').
                                replace('!',''))
df.sample(5)
```

	listing_id	id	date	reviewer_id	reviewer_name	comments
366332	21364988	267741008	2018-05-22	44288049	Ella	Staying at Kennedy's was a pleasure He welcome...
241644	17734371	285001752	2018-07-02	1936664	Mathew	Because I am traveling with my wife 2 little g...
447923	6795657	324664475	2018-09-17	88398869	Sean	Great place
276561	18793633	318216075	2018-09-03	26334457	Oliver	Our stay with Lauren was amazing and exactly w...
193780	16288167	273566222	2018-06-06	173858533	Stefano	Ottima posizione per raggiungere sia l'aeropor...

```
# 停用词
stopwords=["'d", "'ll", "'m", "'re", "'s", "'t", "'ve", "ZT", "ZZ", "a",
           "a's", "able", "about", "above", "abst", "accordance", "according",
           "accordingly", "across", "act", "actually", "added", "adj", "adopted", "affected", "affecting",
```

```
# 对评论的分词函数
def word_split(comment):
    lt = comment.split(' ')
    bool_ = [word in stopwords for word in lt]
    words = []
    for i in range(len(lt)):
        if bool_[i]==False:
            words.append(lt[i])
    return words
```

```
# 应用上述分词函数对'comments'列进行切分，并且将结果存入新的'words'列中
df['words'] = df.comments.apply(word_split)
```

```
# 获取df.words数据列表
words_lt = df.words.to_list()
```

```
# 对于每一条评论分词后的列表进行取词操作，存入word_lt列表
word_lt = []
for temp in words_lt:
    for i in range(len(temp)):
        word_lt.append(temp[i])
```

```
# 计算词频（运行时间特别长，已经将结果写入csv文件）
words_set = set(word_lt)
word_freq = {}
for w in words_set:
    word_freq[w] = word_lt.count(w)
```

```
# 继续清洗词频数据
word_freq_after = {}
for x,y in word_freq.items():
    x=x.lower()
    if x == '' or x.isdigit():
        continue
    if x not in stopwords:
        word_freq_after[x]=y
```

```
# 按词频降序输出
word_freq_sorted = dict(sorted(word_freq.items(),key=lambda kv:kv[1],reverse=True))
#print(word_freq_sorted)
```

绘制词云图

```
# 读入清洗整理好的2018年的评论数据
word_freq_sorted_df = pd.read_csv('word_freq_sorted.csv',index_col=0)
word_freq_sorted_df.head()
```

word	freq
comfortable	56015
space	42044
restaurants	33544
nyc	33251
neighborhood	31032

```
word_freq_sorted_dict = word_freq_sorted_df.to_dict()
word_freq_sorted_dict = word_freq_sorted_dict['freq']
```

```
print('Airbnb网站2018年评论分词词频数据字典长度:',len(word_freq_sorted_dict))
```

```
Airbnb网站2018年评论分词词频数据字典长度: 125688
```

```
#载入词云软件包
from os import path
import imageio
import matplotlib.pyplot as plt
from wordcloud import WordCloud,ImageColorGenerator
```

```
#词云函数
wc = WordCloud(
    font_path='/Windows/Fonts/STKAITI.TTF',
    background_color="white",width=1000,
    height=600,max_words=10000,
    max_font_size=250,random_state=42)
```

```
#导入词频字典格式
wc.generate_from_frequencies(word_freq_sorted_dict)
```

```
<wordcloud.wordcloud.WordCloud at 0x2034502da30>
```

```
#绘制
wc.to_file("2018年Airbnb客户评论词云图.png")
```

```
<wordcloud.wordcloud.WordCloud at 0x2034502da30>
```

```
# 展示词云图
from PIL import Image
Image.open('2018年Airbnb客户评论词云图.png')
```

词云图的视觉冲击效果一流，从上述词云图清晰看出 2018 年 Airbnb 网站评论数据的出现频率最高的是'comfortable','space','restaurants','nyc','neighborhood','bed','enjoyed','hosts', 'communication','excellent'等。

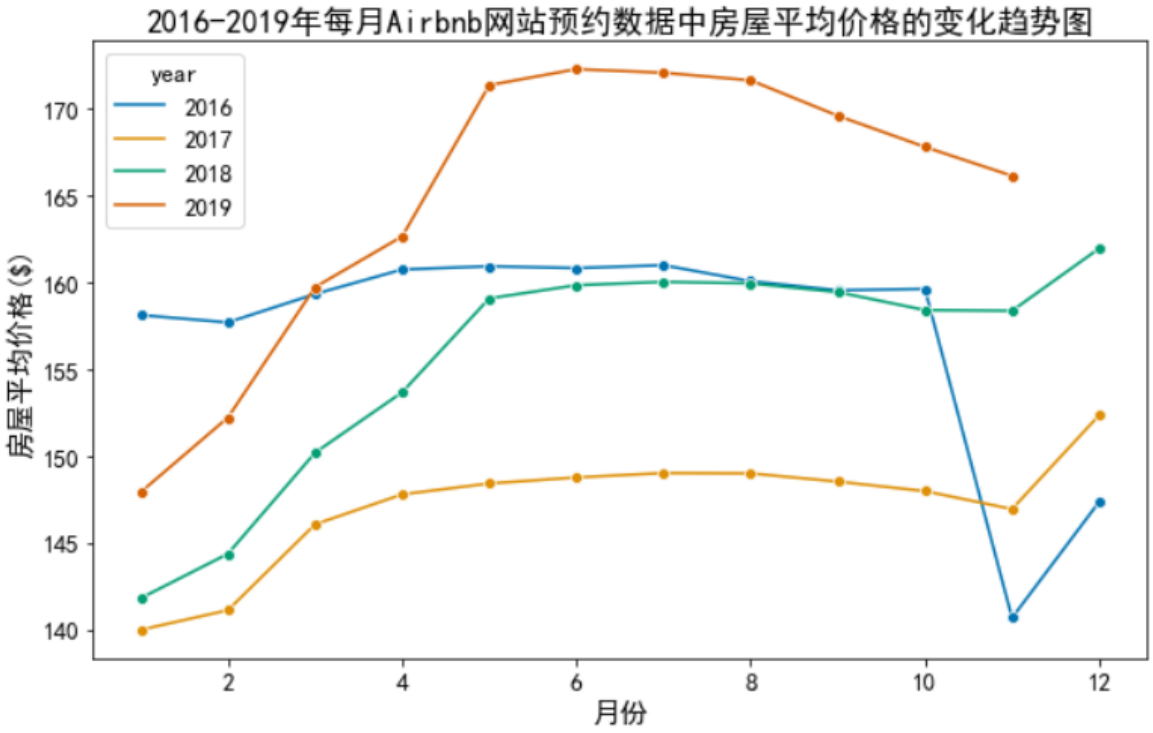
11. 利用 calendar（预约表）中的时间序列绘制每年每月房屋平均价格的变化趋势折线图

```
price_mean = calendar_df.groupby(['year','month'])[['price']].mean()
price_mean.head()
```

year	month	price
2015	11	161.928154
	12	168.427944
2016	1	158.129200
	2	157.709497
	3	159.341177

```
# 由于2015年数据只有11月和12月的，故删除
price_mean = price_mean.drop(2015)
```

```
plt.figure(figsize=(10,6))
sns.lineplot(data=price_mean,x='month',hue='year',y='price',marker='o',palette='colorblind')
plt.ylabel("房屋平均价格($)",fontsize=15)
plt.xlabel('月份',fontsize=15)
plt.title("2016-2019年每月Airbnb网站预约数据中房屋平均价格的变化趋势图",fontsize=17)
plt.show()
```



分析：从上述各年的平均价格折线图可以看出，1-5 月的房屋平均价格总体呈现上升趋势，5-8 月的房屋平均价格呈现趋于平稳，8-11 月价格会有平缓下降的趋势，而后的 11-12 月房屋价格会有明显上升的趋势。

五．总结

1. 纽约各个地区的房源数量分布排名情况依次是：Manhattan(23481)，Brooklyn(20614)，Queens(5515)，Bronx(1003)，StatenIsland(334)；

2. Airbnb 网站上较多房源数量的房屋类型是 Apartment,House,loft,Condominium, Townhouse 和 Serviceapartment。

3. Airbnb 网站上房源数量占比最多的房间类型是 Entirehome/apt 和 Privateroom，Sharedroom 的占比较小；

房源数量占比最多的床类型是 RealBed；而 Pull-outSofa，Airbed 和 Couch 占比都较少，房源数量与 RealBed 的差距较大

4. 需要保证金的房屋数量为:21366;占比为:41.94%；

需要清理费用的房屋数量为:36734;占比为:72.1%；

保证金金额大致集中在 0-1000$之间，其中存在需要大额保证金的部分房屋，最高有需要 5000$的房屋；

清理房屋费用金额大致集中在 0-200$之间，其中同样存在需要较大额清理费用的房屋，最高清理费用可达 1000$左右。

5. 房屋价格主要分布在 0-1000$之间，房屋价格的总体分布是 0-10000$,不过 1000—10000$的房源数量占比极小。

6. 纽约市各地区的房屋平均价格排名情况依次是：

Manhattan(192.76$),StatenIsland(132.94$),Brooklyn(121.53$)，Queens(97.26)，Bronx(86.04$)；Manhattan 的平均价格最高，Bronx 平均价格最低

7. 房屋评分和价格的相关性情况

评分的分布区间主要分布在 60-100 之间，从 scipy 的曲线拟合情况来看，房屋价格和评分的相关性大致呈现指数递增趋势，但并不是指数爆炸型增长，增长趋势较为平缓。

8. 从房屋数量，建房年数，评论数据，评分，房屋价格五个维度评估纽约市五个地区的房源情况；得出各地区的房屋情况评估排名是：

Manhattan, Brooklyn, Queens, Bronx,StatenIsland；但 Bronx 和 StatenIsland 的差距并不明显。

9. 房屋各项指标的热力图得出各项之间的相关性：

与价格相关性较为明显的是卧室数量，清理费用和保证金；和清理费用相关性较为明显的是卧室数量，保证金和价格；和评分相关性较为明显的是评论数量和清理费用。

10. 2018 年 Airbnb 网站评论数据的出现频率最高的单词是'comfortable','space','restaurants','nyc','neighborhood','bed','enjoyed','hosts','communication','excellent'等，基本上都是正向或者中性评论情感词。

11. Airbnb 网站上房源平均价格的总体变化趋势是：

1-5 月的房屋平均价格总体呈现上升趋势，5-8 月的房屋平均价格呈现趋于平稳，8-11 月价格会有平缓下降的趋势，而后的 11-12 月房屋价格会有明显上升的趋势。

六．课程设计体会

在这个学期里，我学习了数据可视化技术，这是一门将数据与视觉表现相结合的学科。学习了如何使用 Python 编程语言来优化数据分析过程。这样我就可以在不同的阶段使用不同的工具来减少重复的工作，提高工作效率。通过期末的课程设计明显地体会到将复杂的数据转化为直观的图表，帮助我们更好地理解数据背后的信息，提供了更高效的数据分析方式。其中的重要之处是学会了如何选择合适的图表类型来呈现数据。

通过学习这门课程，我收获了很多宝贵的经验，在此深表对于老师的感激之情。我掌握了更多有关数据分析和图表制作的技巧，而这些技能将在我未来的工作中起到重要的作用，让我深刻体会到了数据可视化技术的魅力。在未来的学习和工作中，我将继续深入研究数据可视化技术，探索更多的应用场景，不断提升自己的技能和水平。