酒店预定项目分析

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt

# 绘图中设置中文字体
from matplotlib import font_manager
my_font = font_manager.FontProperties(fname="C:/Windows/Fonts/simsun.ttc")
```

```
# 导入数据
file_path = "./hotel_bookings.csv"
df = pd.read_csv(file_path)
```

数据预处理

```
# 查看前五条数据
df.head()
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_
0	Resort Hotel	0	342	2015	July	27	1
1	Resort Hotel	0	737	2015	July	27	1
2	Resort Hotel	0	7	2015	July	27	1
3	Resort Hotel	0	13	2015	July	27	1
4	Resort Hotel	0	14	2015	July	27	1

5 rows × 32 columns

```
# 查看表信息
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   hotel                           119390 non-null  object 
 1   is_canceled                     119390 non-null  int64  
 2   lead_time                       119390 non-null  int64  
 3   arrival_date_year               119390 non-null  int64  
 4   arrival_date_month              119390 non-null  object 
 5   arrival_date_week_number        119390 non-null  int64  
 6   arrival_date_day_of_month       119390 non-null  int64  
 7   stays_in_weekend_nights         119390 non-null  int64  
 8   stays_in_week_nights            119390 non-null  int64  
 9   adults                          119390 non-null  int64  
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64  
 12  meal                            119390 non-null  object 
 13  country                         118902 non-null  object 
 14  market_segment                  119390 non-null  object 
 15  distribution_channel            119390 non-null  object 
 16  is_repeated_guest               119390 non-null  int64  
 17  previous_cancellations          119390 non-null  int64  
 18  previous_bookings_not_canceled  119390 non-null  int64  
 19  reserved_room_type              119390 non-null  object 
 20  assigned_room_type              119390 non-null  object 
 21  booking_changes                 119390 non-null  int64  
 22  deposit_type                    119390 non-null  object 
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64  
 26  customer_type                   119390 non-null  object 
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64  
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

```
# 查看有多少条数据
df.shape()
```

```
(119390, 32)
```

```
pd.DataFrame(df.isnull().sum(), columns=["缺失值数量"])
```

	缺失值数量
hotel	0
is_canceled	0
lead_time	0
arrival_date_year	0
arrival_date_month	0
arrival_date_week_number	0
arrival_date_day_of_month	0
stays_in_weekend_nights	0
stays_in_week_nights	0
adults	0
children	4
babies	0
meal	0
country	488
market_segment	0
distribution_channel	0
is_repeated_guest	0
previous_cancellations	0
previous_bookings_not_canceled	0
reserved_room_type	0
assigned_room_type	0
booking_changes	0
deposit_type	0
agent	16340
company	112593
days_in_waiting_list	0
customer_type	0
adr	0
required_car_parking_spaces	0
total_of_special_requests	0
reservation_status	0
reservation_status_date	0

数据清洗

```
# 空值替换
nan_replacements = {"children": 0.0,"country": "Unknown", "agent": 0, "company": 0}
df.fillna(nan_replacements, inplace=True)

# "meal"中"Undefined","SC"没什么区别，把"Undefined"替换成"SC"
df["meal"].replace("Undefined", "SC", inplace=True)

#有些数据中，婴儿、小孩、大人数量都为0，这些数据删除
zero_guests = list(df.loc[df["adults"]
                   + df["children"]
                   + df["babies"]==0].index)
df.drop(df.index[zero_guests], inplace=True)
```

```
# 查看还剩多少条数据
df.shape
```

```
(119210, 32)
```

```
pd.DataFrame(df.isnull().sum(), columns=["缺失值数量"])
```

	缺失值数量
hotel	0
is_canceled	0
lead_time	0
arrival_date_year	0
arrival_date_month	0
arrival_date_week_number	0
arrival_date_day_of_month	0
stays_in_weekend_nights	0
stays_in_week_nights	0
adults	0
children	0
babies	0
meal	0
country	0
market_segment	0
distribution_channel	0
is_repeated_guest	0
previous_cancellations	0
previous_bookings_not_canceled	0
reserved_room_type	0
assigned_room_type	0
booking_changes	0
deposit_type	0
agent	0
company	0
days_in_waiting_list	0
customer_type	0
adr	0
required_car_parking_spaces	0
total_of_special_requests	0
reservation_status	0
reservation_status_date	0

数据分析

基本情况：

- 酒店客户分别来自于哪个国家；
- 城市酒店和假日酒店预定入住率比较；

用户行为：提前预订时长、入住时长、预订间隔、餐食预订情况；

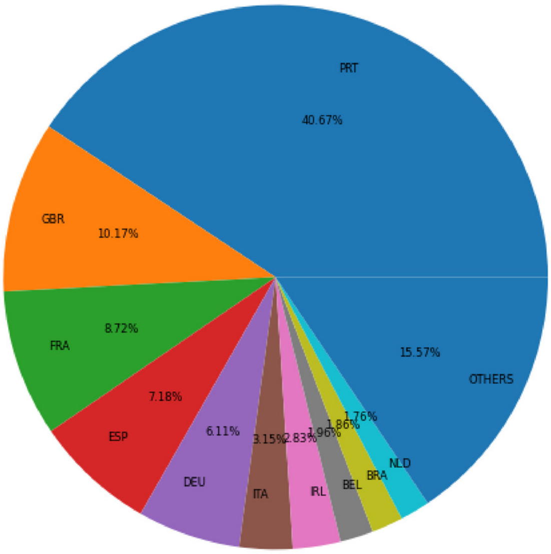
- 客户提前几天预定
- 客户在酒店住几天

预定建议：

- 哪个月酒店最忙
- 一年中每晚价格变化是怎样的

1.1酒店客户分别来自于哪个国家

```
country_data = df.groupby(by="country")["hotel"].count().sort_values(ascending=False)

# 取数量前十的城市，其余合并为others
country_data_num = list(country_data.values)[0:10]
country_data_num.append(sum(list(country_data.values)[10:]))

country_data_name = list(country_data.index)[0:10]
country_data_name.append("OTHERS")

# 画图
plt.figure(figsize=(15,10), dpi=60)

labels = country_data_name
sizes = country_data_num

plt.pie(sizes, labels=labels, labeldistance=0.8, autopct = '%3.2f%%')

# x, y轴刻度设置一致，保证饼图为圆形
plt.axis('equal')

plt.show()
```



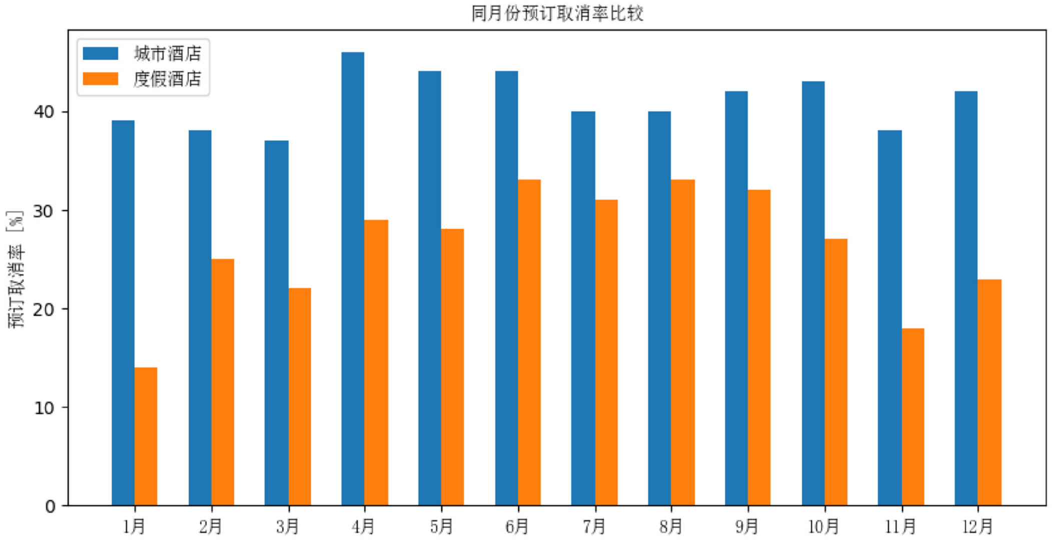
入住这两家酒店客人中，数量最多的来自于：PRT葡萄牙、GBR英国、FRA法国、ESP西班牙、DEU德国。客人绝大多数都来自于欧洲，来自葡萄牙的客人最多，所以我们基本可以确定，这份数据来自于葡萄牙。

1.2城市酒店和假日酒店入住率比较

```
# 取消预订的数,取消预订的字段值为1
sum_cancel = df["is_canceled"].sum()
rh_cancel = df.loc[df["hotel"] == "Resort Hotel"]["is_canceled"].sum()
ch_cancel = df.loc[df["hotel"] == "City Hotel"]["is_canceled"].sum()

sum_cancel_percent = sum_cancel / df["is_canceled"].count() * 100
rh_cancel_percent = rh_cancel / df.loc[df["hotel"] == "Resort Hotel"]["hotel"].count() * 100
ch_cancel_percent = ch_cancel / df.loc[df["hotel"] == "City Hotel"]["hotel"].count() * 100

print("总体取消预订数量为：%d 占比：%2.2f%%" % (sum_cancel, sum_cancel_percent))
print("度假酒店取消预订数量为：%d 占比：%2.2f%%" % (rh_cancel, rh_cancel_percent))
print("城市酒店取消预订数量为：%d 占比：%2.2f%%" % (ch_cancel, ch_cancel_percent))
```

```
总体取消预订数量为：44199 占比：37.08%
度假酒店取消预订数量为：11120 占比：27.77%
城市酒店取消预订数量为：33079 占比：41.79%
```

```
# 不同月份入住率比较
# 度假酒店按照月份分组
rh_cancel_bymon = df.loc[df["hotel"] == "Resort Hotel"].groupby(by="arrival_date_month")["is_canceled"].sum()
ch_cancel_bymon = df.loc[df["hotel"] == "City Hotel"].groupby(by="arrival_date_month")["is_canceled"].sum()

rh_all_bymon = df.loc[df["hotel"] == "Resort Hotel"].groupby(by="arrival_date_month")["is_canceled"].count()
ch_all_bymon = df.loc[df["hotel"] == "City Hotel"].groupby(by="arrival_date_month")["is_canceled"].count()

# 画图
plt.figure(figsize=(10,5), dpi=100)
plt.title("同月份预订取消率比较", fontproperties=my_font)

# 根据实际月份顺序对应
_x = [4,8,12,2,1,7,6,3,5,11,10,9]

rh_y = []
ch_y = []
for i in list(range(12)):
    rh_y.append(int(rh_cancel_bymon[i] / rh_all_bymon[i] * 100))
    ch_y.append(int(ch_cancel_bymon[i] / ch_all_bymon[i] * 100))

plt.bar(_x,ch_y, width=0.3, label="城市酒店")
plt.bar([i+0.3 for i in _x],rh_y, width=0.3, label="度假酒店")

plt.xticks([i+0.15 for i in (range(1,13))],[str(i)+"月" for i in list(range(1,13))], fontproperties=my_font)
plt.ylabel("预订取消率 [%]", fontproperties=my_font)

plt.legend(loc="best", prop=my_font)
plt.show()
```

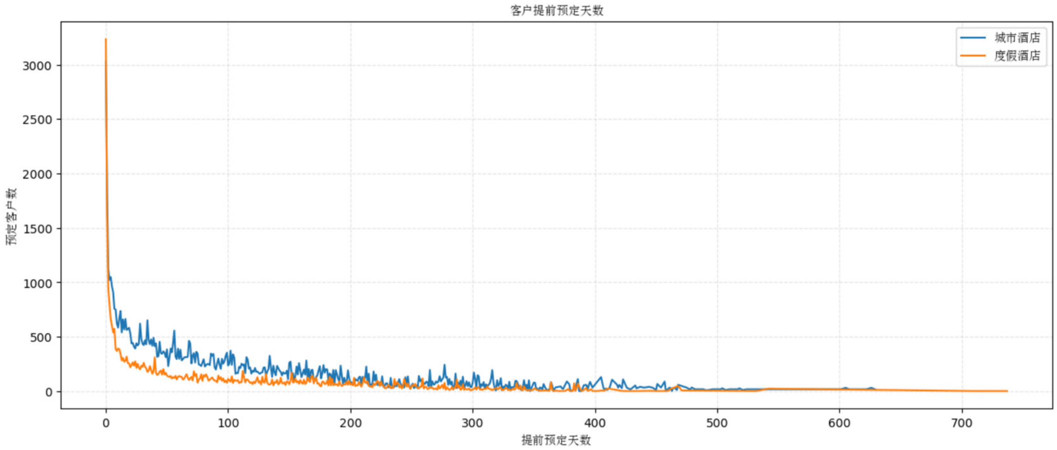


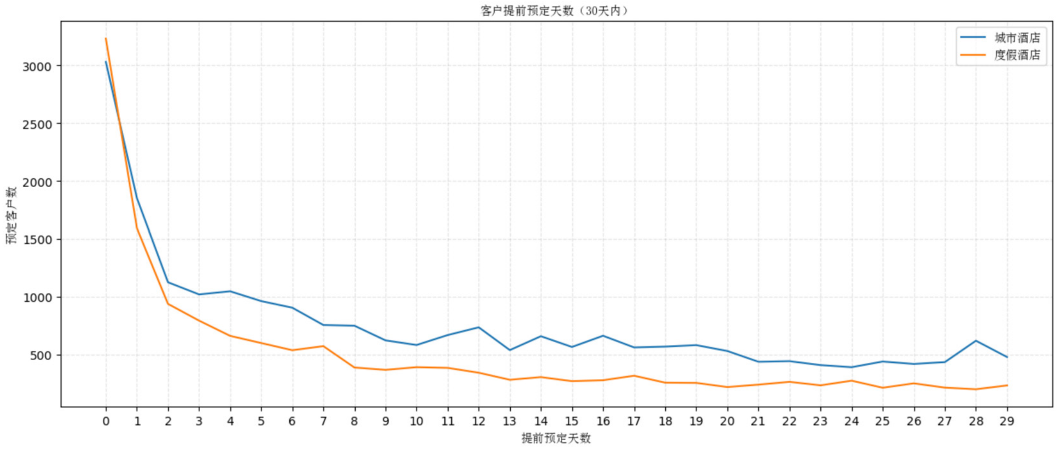
城市酒店预订取消率整体高于度假酒店，这与酒店性质有关，旅游行程的改变概率一般比较小；城市酒店全年取消率较为稳定，在40%左右浮动；度假酒店每年冬天取消率比较低，夏天比较高。

2.1客户提前几天预定

```
lead_time = df.groupby(by=["hotel","lead_time"])["lead_time"].count()

# 画图1
plt.figure(figsize=(15,6), dpi=100)
plt.title("客户提前预定天数", fontproperties=my_font)
```

```
ch_x = list(lead_time.loc["City Hotel"].index)
rh_x = list(lead_time.loc["Resort Hotel"].index)

ch_y = list(lead_time.loc["City Hotel"].values)
rh_y = list(lead_time.loc["Resort Hotel"].values)

plt.plot(ch_x, ch_y, label="城市酒店")
plt.plot(rh_x, rh_y, label="度假酒店")

plt.grid(alpha=0.3, linestyle='--')

plt.xlabel("提前预定天数", fontproperties=my_font)
plt.ylabel("预定客户数", fontproperties=my_font)

plt.legend(loc="best", prop=my_font)
plt.show()


# 画图2
plt.figure(figsize=(15,6), dpi=100)
plt.title("客户提前预定天数（30天内）", fontproperties=my_font)


ch_x = list(lead_time.loc["City Hotel"].index)[0:30]
rh_x = list(lead_time.loc["Resort Hotel"].index)[0:30]

ch_y = list(lead_time.loc["City Hotel"].values)[0:30]
rh_y = list(lead_time.loc["Resort Hotel"].values)[0:30]

plt.plot(ch_x, ch_y, label="城市酒店")
plt.plot(rh_x, rh_y, label="度假酒店")


plt.xticks(ch_x, list(range(30)))

plt.grid(alpha=0.3, linestyle='--')

plt.xlabel("提前预定天数", fontproperties=my_font)
plt.ylabel("预定客户数", fontproperties=my_font)

plt.legend(loc="best", prop=my_font)
plt.show()
```

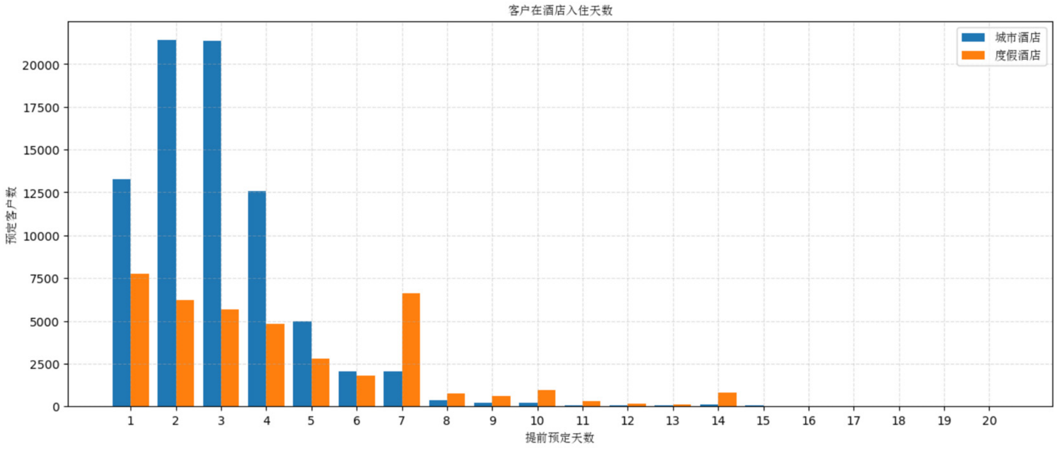




第一张图来看，提前预定相同天数的情况下，城市酒店预定的客户数要比旅游酒店多，但是预定城市酒店的总人数较多，所以整体趋势两者基本是相同的；第一张图可以看出，提前很少的天数预定的人数最多，然后断崖式下降，随着预定天数的增加，预定的人数也逐步减少，但没有再出现突然断崖式降低的情况；截取第一张图的前30个数据，绘制出第二张图，可以看出，提前预定0天、1天、2天的客户数剧烈下降，提前2天以后预定的客户数量则平缓下降。

2.2客户在酒店住几天

```
df["stay_time"] = df["stays_in_weekend_nights"] + df["stays_in_week_nights"]

stay_time = df.groupby(by=["hotel","stay_time"])["stay_time"].count()

# 画图1
plt.figure(figsize=(15,6), dpi=100)
plt.title("客户在酒店入住天数", fontproperties=my_font)


ch_x = list(stay_time.loc["City Hotel"].index)[1: 21]
rh_x = list(stay_time.loc["Resort Hotel"].index)[1: 21]

ch_y = list(stay_time.loc["City Hotel"].values)[1: 21]
rh_y = list(stay_time.loc["Resort Hotel"].values)[1: 21]

plt.bar(ch_x, ch_y, width=0.4, label="城市酒店")
plt.bar([i+0.4 for i in rh_x], rh_y, width=0.4, label="度假酒店")

plt.xticks([i+0.2 for i in ch_x], list(range(1,21)))

plt.grid(alpha=0.4, linestyle='--')

plt.xlabel("提前预定天数", fontproperties=my_font)
plt.ylabel("预定客户数", fontproperties=my_font)

plt.legend(loc="best", prop=my_font)
plt.show()

# 计算入住天数平均值
# List3 = np.multiply(np.array(List1),np.array(List2)).tolist() 通过转换成数组，求两个列表对应值的积
sum_stay_ch = sum((np.multiply(np.array(list(stay_time.loc["City
Hotel"].index)),np.array(list(stay_time.loc["City Hotel"].values)))).tolist())
avg_stay_ch = sum_stay_ch / sum(list(stay_time.loc["City Hotel"].values))
max_stay_ch = max(list(stay_time.loc["City Hotel"].index))

sum_stay_rh = sum((np.multiply(np.array(list(stay_time.loc["Resort
Hotel"].index)),np.array(list(stay_time.loc["Resort Hotel"].values)))).tolist())
avg_stay_rh = sum_stay_rh / sum(list(stay_time.loc["Resort Hotel"].values))
max_stay_rh = max(list(stay_time.loc["Resort Hotel"].index))

print("城市酒店客人平均住宿%.2f天，最多住宿%d天" % (avg_stay_ch, max_stay_ch))
print("度假酒店客人平均住宿%.2f天，最多住宿%d天" % (avg_stay_rh, max_stay_rh))
```



```
城市酒店客人平均住宿2.97天，最多住宿48天
度假酒店客人平均住宿4.32天，最多住宿69天
```

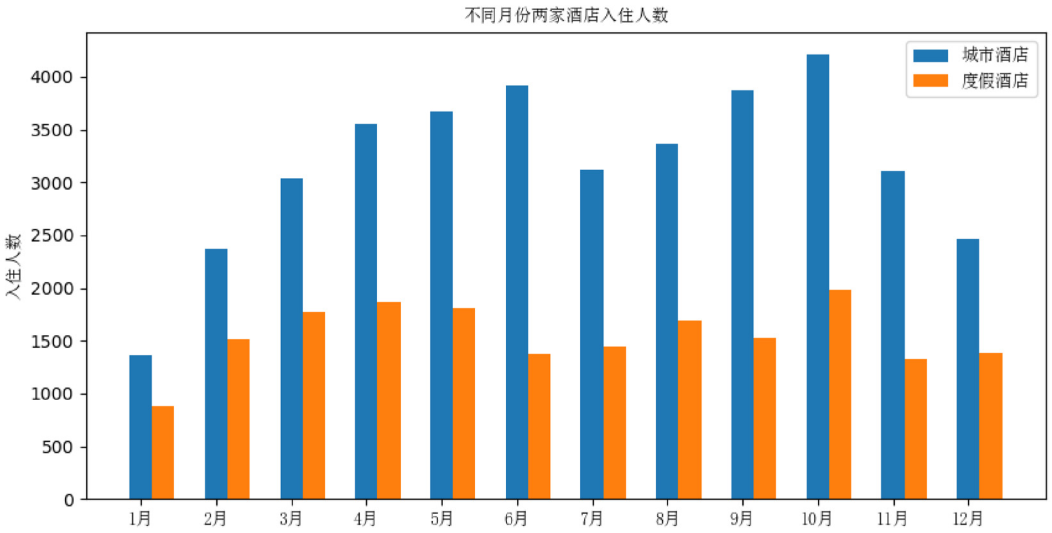
城市酒店大家一般住1-4天比较多，而度假酒店1-6天住宿人数依次减少，而住7天的突然增多，14天也有一个小的增长。

3.1哪个月酒店最忙

```
a = df.loc[df["hotel"] == "Resort Hotel"].groupby(by=["arrival_date_year","arrival_date_month"])
["arrival_date_month"].count()
list(a[2016].values)
```

```
[1867, 1685, 1381, 1519, 884, 1441, 1369, 1778, 1802, 1331, 1984, 1523]
```

```
# 不同月份入住人数比较
# 酒店按照月份分组
rh_arrival_date_bymon = df.loc[df["hotel"] == "Resort Hotel"].groupby(by=
["arrival_date_year","arrival_date_month"])["arrival_date_month"].count()
ch_arrival_date_bymon = df.loc[df["hotel"] == "City Hotel"].groupby(by=
["arrival_date_year","arrival_date_month"])["arrival_date_month"].count()

# 画图
plt.figure(figsize=(10,5), dpi=100)
plt.title("不同月份两家酒店入住人数", fontproperties=my_font)

# 根据实际月份顺序对应
_x = [4,8,12,2,1,7,6,3,5,11,10,9]

# 由于只有2016年包含所有月份，选取2016年数据
rh_y = list(rh_arrival_date_bymon[2016].values)
ch_y = list(ch_arrival_date_bymon[2016].values)

plt.bar(_x,ch_y, width=0.3, label="城市酒店")
plt.bar([i+0.3 for i in _x],rh_y, width=0.3, label="度假酒店")


plt.xticks(list(range(1,13)),[str(i)+"月" for i in list(range(1,13))], fontproperties=my_font)

plt.ylabel("入住人数", fontproperties=my_font)
plt.legend(loc="best", prop=my_font)

plt.show()
```

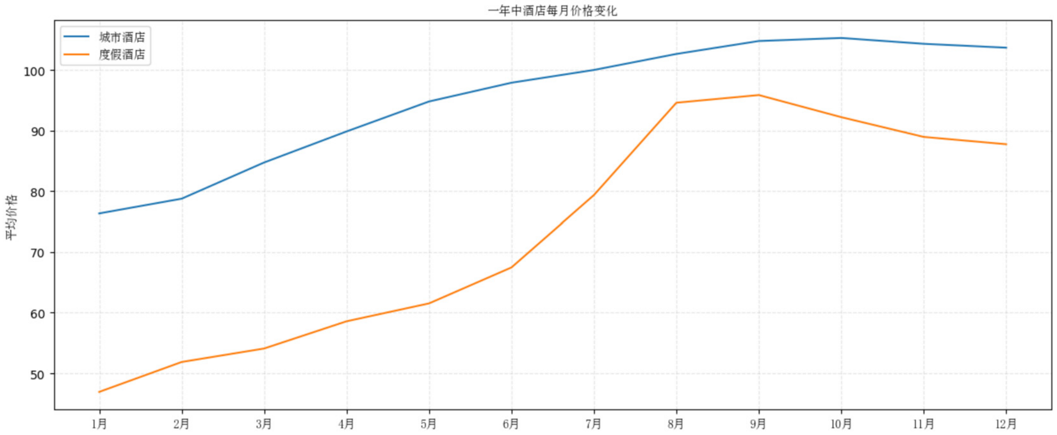


两家酒店变化趋势基本相同，春秋两季入住人数较多，夏天和冬季人数较少，所以可以预测，春季和冬季酒店住宿价格会较高，接下来我会分析价格波动，看看预测是否准确。

3.2一年中酒店每月价格变化是怎样的

```
# 城市酒店
ch_adr_group_sum = df.loc[df["hotel"] == "City Hotel"].groupby(by=
["arrival_date_year","arrival_date_month","arrival_date_day_of_month"])["adr"].sum()
ch_adr_group_count = df.loc[df["hotel"] == "City Hotel"].groupby(by=
["arrival_date_year","arrival_date_month","arrival_date_day_of_month"])["adr"].count()

order_month = ["January", "February", "March", "April", "May", "June", "July", "August", "September",
"October", "November", "December"]
ch_adr_month_sum = 0
ch_adr_month_count = 0
ch_adr_month_avg=[]
for i in order_month:
    for ii in ch_adr_group_sum[2016,i].values:
        ch_adr_month_sum += ii
    for ii in ch_adr_group_count[2016,i].values:
        ch_adr_month_count += ii
    ch_adr_month_avg.append(ch_adr_month_sum/ch_adr_month_count)

# 度假酒店
rh_adr_group_sum = df.loc[df["hotel"] == "Resort Hotel"].groupby(by=
["arrival_date_year","arrival_date_month","arrival_date_day_of_month"])["adr"].sum()
rh_adr_group_count = df.loc[df["hotel"] == "Resort Hotel"].groupby(by=
["arrival_date_year","arrival_date_month","arrival_date_day_of_month"])["adr"].count()
```

```
order_month = ["January", "February", "March", "April", "May", "June", "July", "August", "September",
"October", "November", "December"]
rh_adr_month_sum = 0
rh_adr_month_count = 0
rh_adr_month_avg=[]
for i in order_month:
    for ii in rh_adr_group_sum[2016,i].values:
        rh_adr_month_sum += ii
    for ii in rh_adr_group_count[2016,i].values:
        rh_adr_month_count += ii
    rh_adr_month_avg.append(rh_adr_month_sum/rh_adr_month_count)

# 画图
plt.figure(figsize=(15,6), dpi=100)
plt.title("一年中酒店每月价格变化", fontproperties=my_font)

_x = list(range(1,13))

ch_y = ch_adr_month_avg
rh_y = rh_adr_month_avg

plt.plot(_x, ch_y, label="城市酒店")
plt.plot(_x, rh_y, label="度假酒店")

plt.xticks(list(range(1,13)),[str(i)+"月" for i in list(range(1,13))], fontproperties=my_font)
plt.ylabel("平均价格", fontproperties=my_font)
plt.grid(alpha=0.3, linestyle='--')
plt.legend(loc="best", prop=my_font)

plt.show()
```



城市酒店整体价格高于度假酒店，城市酒店每年价格最高的时间是九月和十月，度假酒店则在八月和九月