数据挖掘第三次作业

题目描述

Hotel booking demand 酒店预订需求

该数据集32列共12W数据，包含城市酒店和度假酒店的预订信息，包括预订时间、停留时间，成人/儿童/婴儿人数以及可用停车位数量等信息。

具体要求：

1、基本情况：城市酒店和假日酒店预订需求和入住率比较；

2、用户行为：提前预订时间、入住时长、预订间隔、餐食预订情况；

3、一年中最佳预订酒店时间；

4、利用Logistic预测酒店预订。

仓库地址

[https://github.com/annwfsly/DataMiningHomework3 (https://github.com/annwfsly/DataMiningHomework3)](https://github.com/annwfsly/DataMiningHomework3)

In [4]:

```
import os
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split

# ======== 数据挖掘第三次作业：Hotel booking demand =========
file_path = "./hotel_bookings.csv"
```

In [5]:

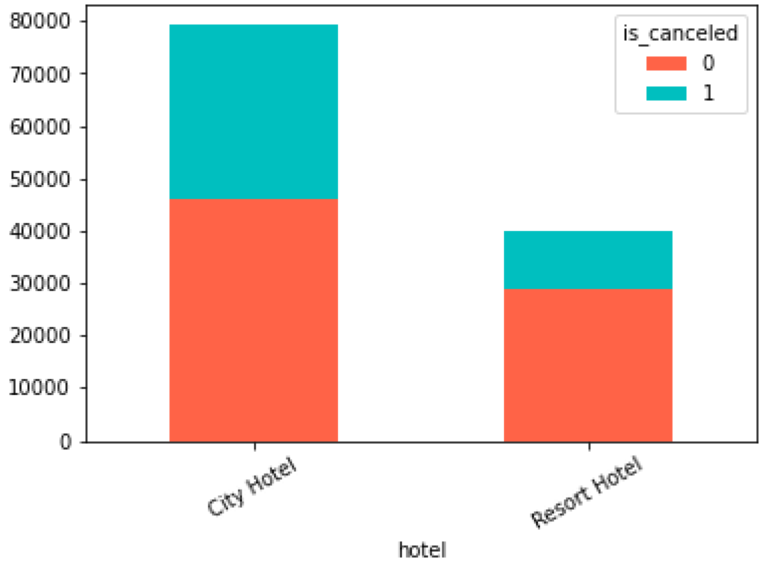
```
# ======== 原始数据初步分析及预处理 ==========
class data_process:
    def __init__(self, dataset):
        self.dataset = dataset

    # ========= 原始数据信息查看、数据清洗 =========
    def read_info_process(self):
        data = dataset.copy()
        # ===== 信息展示 =====
        print("=========Info Print:=============")
        data.info()
        # ===== 缺失值处理 ======
        print("===========Lost Data:============")
        print(data.isnull().sum().sort_values(ascending=False))
        '''
        缺失值:
        company   112593  --> 补0
        agent     16340   --> 补0
        country   488     --> 补众数
        children  4       --> 补均值
        '''

        data = data.drop(data[(data.adults+data.children+data.babies)==0].index)
        data[['agent', 'company']] = data[['agent', 'company']].fillna(0)
        data['children'].fillna(round(data.children.mean()), inplace=True)
        data[['children', 'company', 'agent']] = data[['children', 'company', 'agent']].astype('int6
        data['country'].fillna(data.country.mode().to_string(), inplace=True)
        # data['arrival_date_month'] = data['arrival_date_month'].map({'January': 1, 'February': 2,
        #                                                              'April': 4, 'May': 5, 'June':
        #                                                              'August': 8, 'September': 9,
        #                                                              'November': 11, 'December': 1
        self.booking_info(data)
        self.best_booktime(data)
        self.pred_model(self.dataset)

    # ======== 酒店预订情况 =========
    def booking_info(self, data):
        # ===== 城市酒店和假日酒店预订需求和入住率比较 =====
        data_c = data[['hotel', 'is_canceled', 'adr']]
        pivot_df = pd.pivot_table(data_c, values='adr', index='hotel', columns='is_canceled', aggfun
        print(pivot_df)
        # === stacked默认False，并列条形图，True改为堆积条形图 ===
        pivot_df.plot.bar(stacked=True, color=['tomato', 'c'])
        plt.xticks(rotation=30)
        plt.show()

        # ==== 用户提前预订时间 =====
        data_lead_time = data[['hotel', 'lead_time']]
        box_resort_lead = data_lead_time[data_lead_time['hotel'] == 'Resort Hotel']
        box_city_lead = data_lead_time[data_lead_time['hotel'] == 'City Hotel']
        self.draw_box(box_resort_lead['lead_time'], 'Resort Hotel', 'lead_time')
        self.draw_box(box_city_lead['lead_time'], 'City Hotel', 'lead_time')
        # ===== 用户入住时长 =======
        data_stay_week = data[['hotel', 'stays_in_week_nights']]
        box_resort_week = data_stay_week[data_stay_week['hotel'] == 'Resort Hotel']
        box_city_week = data_stay_week[data_stay_week['hotel'] == 'City Hotel']
        self.draw_box(box_resort_week['stays_in_week_nights'], 'Resort Hotel', 'stays_in_week_nights
        self.draw_box(box_city_week['stays_in_week_nights'], 'City Hotel', 'stays_in_week_nights')
        data_stay_weekend = data[['hotel', 'stays_in_weekend_nights']]
        box_resort_weekend = data_stay_weekend[data_stay_weekend['hotel'] == 'Resort Hotel']
        box_city_weekend = data_stay_weekend[data_stay_weekend['hotel'] == 'City Hotel']
        self.draw_box(box_resort_weekend['stays_in_weekend_nights'], 'Resort Hotel', 'stays_in_weeke
        self.draw_box(box_city_weekend['stays_in_weekend_nights'], 'City Hotel', 'stays_in_weekend_n

        # ===== 预订间隔、餐食预订情况 ====
        data_reserve = data[['hotel', 'reservation_status_date']]
        box_resort_reser = data_reserve[data_reserve['hotel'] == 'Resort Hotel']
        box_city_reser = data_reserve[data_reserve['hotel'] == 'City Hotel']
        data_meal = data[['hotel', 'meal']]
        resort_x, resort_y = self.get_count(data_meal[data_meal['hotel'] == 'Resort Hotel']['meal'])
        plt.bar(resort_x, resort_y)
        plt.title('Resort Hotel meal')
        plt.show()

        city_x, city_y = self.get_count(data_meal[data_meal['hotel'] == 'City Hotel']['meal'])
        plt.bar(city_x, city_y)
        plt.title('City Hotel meal')
        plt.show()

    def draw_box(self, box_x, title, x_lbl):
        plt.boxplot(box_x)
        plt.title(title)
        plt.xlabel(x_lbl)
        plt.show()

    def get_count(self, series, limit=None):
        if limit is not None:
            series = series.value_counts()[:limit]
        else:
            series = series.value_counts()
        x = series.index
        y = series / series.sum() * 100
        return x.values, y.values

    def best_booktime(self, data_new):
        # ===== 每月人均平均每晚价格 =====
        data_new["adr_pp"] = data_new["adr"] / (data_new["adults"] + data_new["children"] + data_new
        full_data_guests = data_new.loc[data_new["is_canceled"] == 0]
        room_price_monthly = full_data_guests[["hotel", "arrival_date_month", "adr_pp"]].sort_values
            "arrival_date_month")
        ordered_months = ["January", "February", "March", "April", "May", "June", "July", "August",
                          "September", "October", "November", "December"]
        month_che = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "D
        for en, che in zip(ordered_months, month_che):
            room_price_monthly["arrival_date_month"].replace(en, che, inplace=True)
        room_price_monthly["arrival_date_month"] = pd.Categorical(room_price_monthly["arrival_date_m
                                                                  categories=month_che, ordered=Tru
        room_price_monthly["hotel"].replace("City Hotel", "City Hotel", inplace=True)
        room_price_monthly["hotel"].replace("Resort Hotel", "Resort Hotel", inplace=True)

        plt.figure(figsize=(12, 8))
        sns.lineplot(x="arrival_date_month", y="adr_pp", hue="hotel", data=room_price_monthly,
                     hue_order=["City Hotel", "Resort Hotel"],
                     ci="sd", size="hotel", sizes=(2.5, 2.5))
        plt.title("month price for one night per man", fontsize=16)
        plt.xlabel("month", fontsize=16)
        plt.ylabel("price for one night per man", fontsize=16)
        plt.show()

    def pred_model(self, dataset):
        # ===== 使用原数据集减少预处理中某些过多缺失值的改变对数据产生影响 =====
        new_data = dataset.copy()[['required_car_parking_spaces', 'lead_time', 'booking_changes',
                                 'adr', 'adults', 'is_canceled']]
        x = new_data.drop(['is_canceled'], axis=1)
        y = new_data['is_canceled']
        # =========== 训练集测试集按比例划分 =====
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, shuffle=False)
        logistic = LogisticRegression()
        logistic.fit(x_train, y_train)
        print("====== train score: " + str(logistic.score(x_train, y_train)))
        # ====== 打印模型参数 ======
        print("====== coef: " + str(logistic.coef_))
        print("====== intercept: " + str(logistic.intercept_))
        # ====== 预测 =======
        y_pred = logistic.predict(x_test)
        print("====== test score: " + str(logistic.score(x_test, y_test)))
        # ====== 模型评价 ======
        confu = confusion_matrix(y_test, y_pred)
        sns.heatmap(confu, annot=True, cmap='YlGnBu')
        plt.title("Confusion Matrix Heatmap")
        plt.show()
```

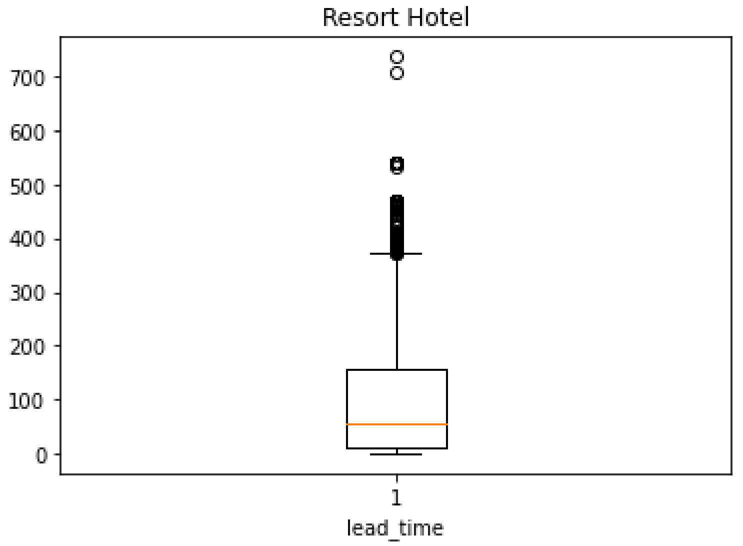
In [6]:

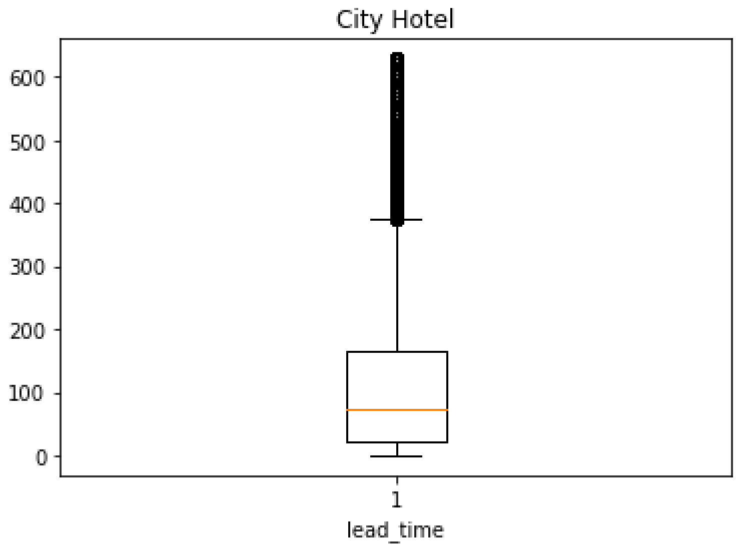
```
if __name__ == "__main__":
    dataset = pd.read_csv(file_path)
    data_process(dataset).read_info_process()
```

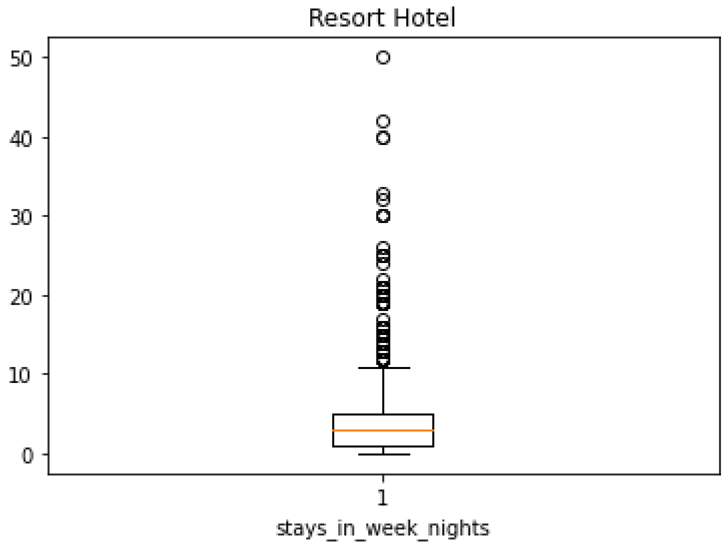
```
=========Info Print:=============
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           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
===========Lost Data:============
company                           112593
agent                              16340
country                              488
children                               4
lead_time                              0
arrival_date_year                      0
arrival_date_month                     0
arrival_date_week_number               0
is_canceled                            0
market_segment                         0
arrival_date_day_of_month              0
stays_in_weekend_nights                0
stays_in_week_nights                   0
```

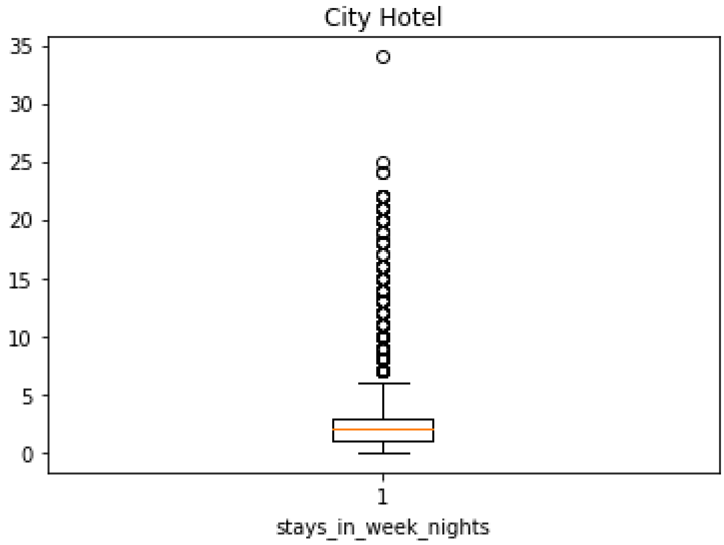
```
adults                            0
babies                            0
meal                              0
reservation_status_date           0
distribution_channel              0
reservation_status                0
is_repeated_guest                 0
previous_cancellations            0
previous_bookings_not_canceled    0
reserved_room_type                0
assigned_room_type                0
booking_changes                   0
deposit_type                      0
days_in_waiting_list              0
customer_type                     0
adr                               0
required_car_parking_spaces       0
total_of_special_requests         0
hotel                             0
dtype: int64
is_canceled       0      1
hotel
City Hotel     46084  33079
Resort Hotel   28927  11120
```

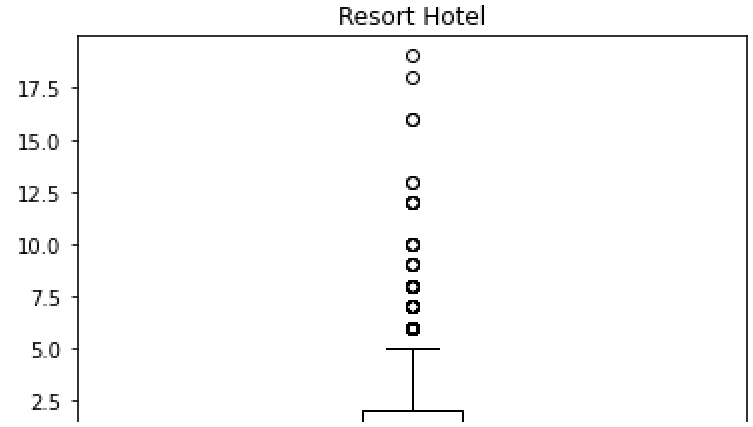


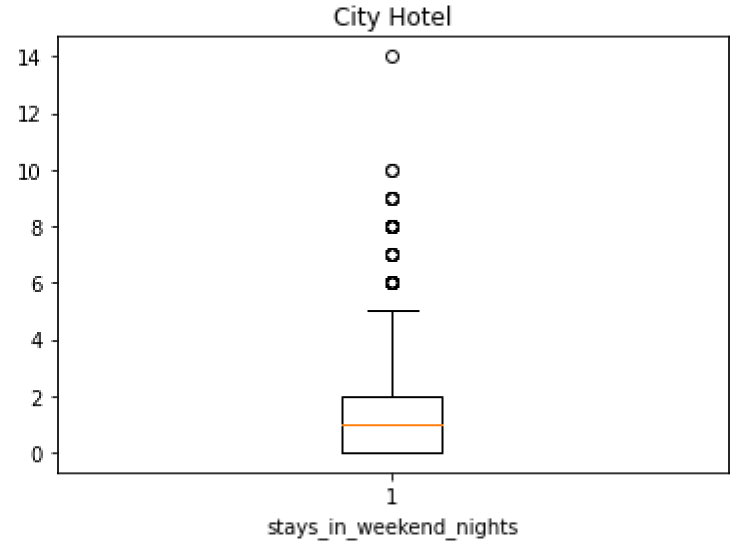


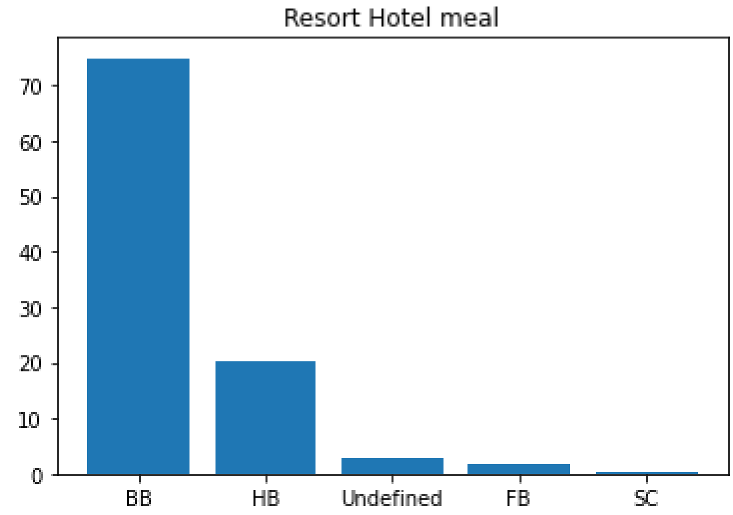

City Hotel
600
500
400
300
200
100
0
1
lead_time

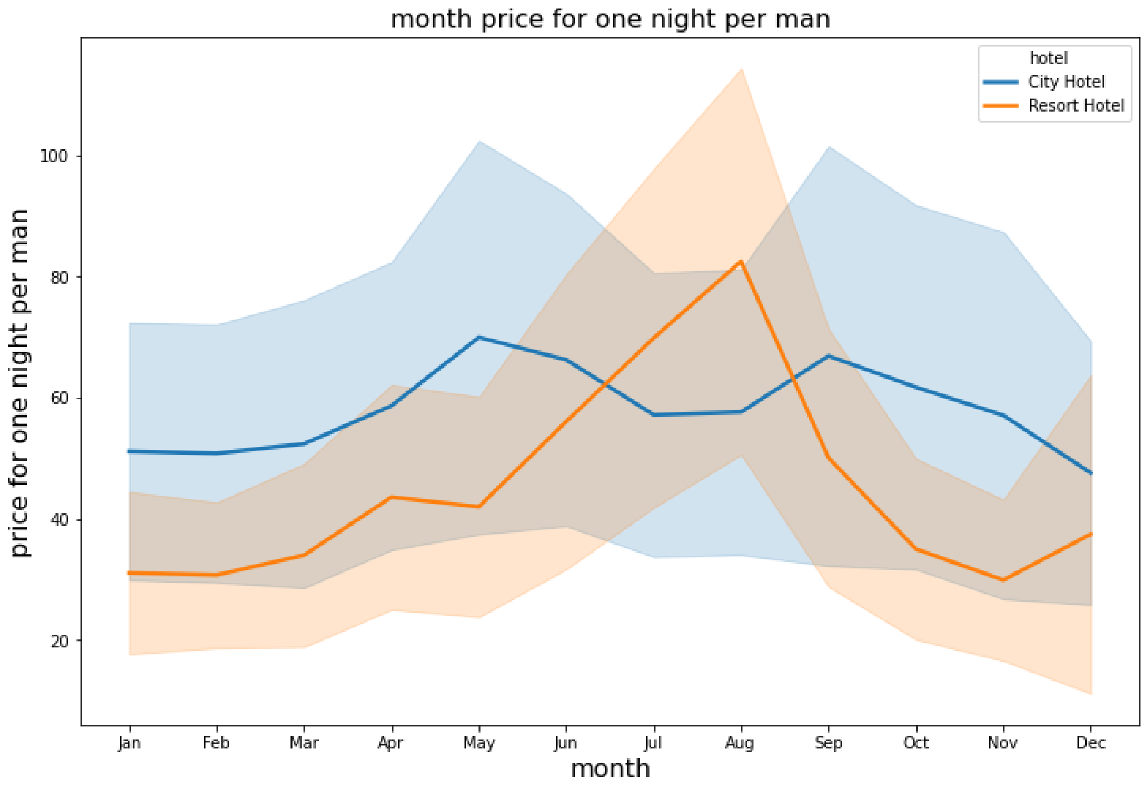

Resort Hotel
50
40
30
20
10
0
1
stays_in_week_nights


City Hotel
35
30
25
20
15
10
5
0
1
stays_in_week_nights

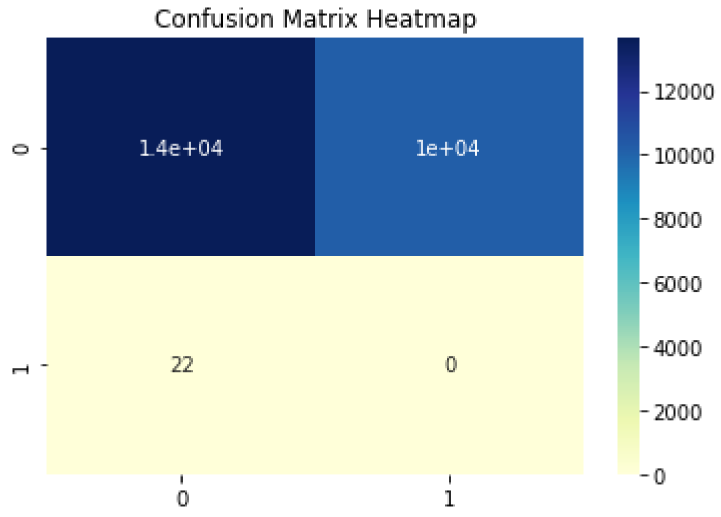

Resort Hotel
17.5
15.0
12.5
10.0
7.5
5.0
2.5


City Hotel
14
12
10
8
6
4
2
0
1
stays_in_weekend_nights


Resort Hotel meal
70
60
50
40
30
20
10
0
BB
HB
Undefined
FB
SC



```
====== train score: 0.6823226400871095
====== coef: [[-6.84306782  0.00701884 -0.75906572  0.00828877 -0.01044419]]
====== intercept: [-1.42809878]
====== test score: 0.5713627607002262
```



In [ ]: