第四組

邱綉雅 林鈺婷 李珩慈 張峻騰

目錄

- 問題描述
- 資料前處理
- 模型介紹
- 資料優化

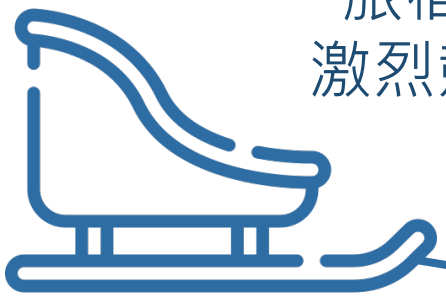
-問題描述-

問題描述- 情境介紹

旅宿業
激烈競爭

彈性
住房方案

空房危機



問題描述-5W1H

藉由Kaggle 上之飯店預訂資料集建立模型，
預測其住房是否取消

What	住房取消預測系統
When	從顧客預訂到入住
Who	旅宿業者
Where	預訂網站、後台
Why	住房取消無法及時重新安排，造成整體住房率降低
How	資料分析、深度學習模型，預測住房是否取消

-資料前處理-

資料前處理-處理空值、資料標準化

```
# check for missing values
full_data.isnull().sum()
```

```
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
```

資料集大小

Size (119210, 32)

共119210筆住宿資料，32個資料欄位

處理空值

```
nan_replacements = {"children:": 0.0,"country": "Unknown", "agent": 0, "company": 0}
full_data_cln = full_data.fillna(nan_replacements)
```

資料標準化

```
df['adr']  =  zscore(df['adr'])
df['lead_time']  =  zscore(df['lead_time'])
```

資料前處理-資料標籤化

資料類別	欄位名稱	資料型態
顧客資料	Adult	數值
	Babies	數值
	Children	數值
	Company	類別
	CustomerType	類別
	Country	類別
	IsRepeatedGuest	類別
	MarketSegment	類別
	PreviousBookingNotCanceled	數值
	PreviousCancellations	數值

資料類別	欄位名稱	資料型態
訂單資料	Agent	類別
	ADR	數值
	ArrivalDateDayOfMonth	數值
	ArrivalDateMonth	類別
	ArrivalDateWeekNumber	數值
	ArrrivalDateYear	數值
	BookingChanges	數值
	DaysInWaitingList	數值
	DepositType	類別
	DistributionChannel	類別
	IsCanceled	類別
	LeadTime	數值
	ReservationStatus	類別
	ReservationStatusDate	日期

資料類別	欄位名稱	資料型態
住房資料	AssignedRoomType	類別
	Meal	類別
	RequiredCarParkingSpaces	數值
	ReservedRoomType	類別
	StaysInWeekendNights	數值
	StaysInWeekNights	數值
	TotalOfSpecialRequestd	數值

資料前處理-資料標籤化

```
df = pd.get_dummies(df, columns = ["required_car_parking_spaces",
                                   "right_room",
                                   "far_in_advance",
                                   "recent_booking",
                                   "changed_booking",
                                   "previous_cancel",
                                   "special_requests",
                                   "arrival_date_year",
                                   "meal",
                                   "market_segment",
                                   "distribution_channel",
                                   "deposit_type",
                                   "customer_type",
                                   "reserved_room_type",
                                   "assigned_room_type"])

le = LabelEncoder()
df['country'] = le.fit_transform(df['country'])
df['hotel'] = le.fit_transform(df['hotel'])

df['arrival_date_month'] = df['arrival_date_month'].map({'January':1, 'February': 2,
                           'March':3, 'April':4, 'May':5, 'June':6, 'July':7,'August':8,
                           'September':9, 'October':10, 'November':11, 'December':12})
```

資料前處理-特徵相關性

```
data_corr = data_corr.drop(['reservation_status','reservation_status_date'], axis=1)
```

資料特徵相關性

lead_time	0.293123
total_of_special_requests	0.234658
required_car_parking_spaces	0.195498
booking_changes	0.144381
previous_cancellations	0.110133

```
cancel_corr = full_data.corr()["is_canceled"]
cancel_corr.abs().sort_values(ascending=False)[1:]
```

```
lead_time                       0.293123
total_of_special_requests       0.234658
required_car_parking_spaces     0.195498
booking_changes                 0.144381
previous_cancellations          0.110133
is_repeated_guest               0.084793
agent                           0.083114
adults                          0.060017
previous_bookings_not_canceled  0.057358
days_in_waiting_list            0.054186
adr                             0.047557
babies                          0.032491
stays_in_week_nights            0.024765
company                         0.020642
arrival_date_year               0.016660
arrival_date_week_number        0.008148
arrival_date_day_of_month       0.006130
children                        0.005048
stays_in_weekend_nights         0.001791
```

-模型介紹-

模型介紹-ML model

```
# define models to test:
base_models = [("DT_model", DecisionTreeClassifier(random_state=42)),
               ("RF_model", RandomForestClassifier(random_state=42,n_jobs=-1)),
               ("LR_model", LogisticRegression(random_state=42,n_jobs=-1)),
               ("XGB_model", XGBClassifier(random_state=42, n_jobs=-1))]
```

	DecisionTree	RandomForest	LogisticRegression	XGBoost
accuracy score	0.8246	0.8664	0.7937	0.8165

模型介紹-NN+

```
#NN+
from  keras.layers  import  Dropout
from  keras.models  import  Sequential
from  keras.layers  import  Dense
model=Sequential()
model.add(Dense(500,input_dim=11,activation='relu'))
model.add(Dropout(0.25))
model.add(Dense(100,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
model.fit(X_train,y_train,epochs=50,batch_size=35,validation_data=(X_test,y_test))
```

模型介紹-NN+

11	500 100 2	sigmoid sigmoid softmax	mean_squared_error	adam	50	35	0.37	0.372
12	500 100 2	sigmoid sigmoid softmax	mean_squared_error	sgd	50	35	0.377	0.378
13	500 100 2	relu relu softmax	mean_squared_error	sgd	50	35	0.37	0.372
14	500 100 2	tanh tanh softmax	mean_squared_error	sgd	50	35	0.369	0.372
15	500 100 2	tanh tanh softmax	mean_squared_error	adam	50	35	0.63	0.628
16	500 100 2	tanh tanh softmax	mean_squared_error	adam	50	70	0.63	0.628
17	500 100 2	tanh tanh softmax	mean_squared_error	adam	50	140	0.37	0.372
18	500 100 32	tanh tanh softmax	mean_squared_error	adam	50	35	0.003	0.003
19	500 100 2	tanh tanh softmax	mean_squared_logarithmic_error	adam	50	35	0.63	0.628
20	500 100 2	relu tanh softmax	mean_squared_logarithmic_error	adam	50	35	0.37	0.372

21	500 100 2	relu tanh softmax	mean_squared_error	adam	50	35	0.63	0.628
22	256 126 64	relu tanh softmax	mean_squared_error	adam	50	35	0	0
23	500 100 20	relu tanh softmax	mean_squared_error	adam	50	35	0	0
24	500 100 2	relu tanh tanh	mean_squared_error	adam	50	35	0.37	0.372
25	500 100 2	relu tanh softmax	mean_squared_error	adam	50	35	0.63	0.628
26	500 100 2	relu tanh softmax	mean_squared_error	adam	100	35	0.63	0.628
27	500 100 2	relu tanh softmax	mean_squared_error	adam	150	35	0.37	0.372
28	500 100 2	relu tanh softmax	mean_squared_error	sgd	50	35	0.382	0.385
29	500 300 2	sigmoid sigmoid softmax	mean_squared_error	sgd	50	35	0.477	0.481
30	500 300 2	relu relu softmax	mean_squared_error	sgd	50	35	0.608	0.606

31	700 300 2	relu relu softmax	mean_squared_error	sgd	50	35	0.512	0.512
32	500 300 2	relu tanh softmax	mean_squared_error	sgd	50	35	0.542	0.538
33	500 300 2	relu tanh softmax	mean_squared_error	adam	50	35	0.37 0.37 0.63	0.372 0.372 0.628
34	500 100 2	relu tanh softmax	mean_squared_error	adam	50	35	0.63	0.627
35	500 100 2	relu relu softmax	mean_squared_error	adam	50	35	0.63	0.628
36	500 100 2	relu relu softmax	mean_squared_error	sgd	50	35	0.629	0.626
37	500 100 2	relu relu softmax	mean_squared_logarithmic_error	sgd	50	35	0.53	0.537
38	300 100 2	relu relu softmax	mean_squared_error	adam	50	35	0.37	0.372
39	300 100 2	relu relu softmax	mean_squared_error	sgd	50	35	0.498	0.494
40	300 100 2	relu relu softmax	mean_squared_error	RMSprop	50	35	0.63	0.628

41	500 100 2	relu relu softmax	mean_squared_error	Adadelta	50	35	0.574	0.572
42	500 100 2	relu relu softmax	mean_squared_error	Adagrad	50	35	0.545	0.547
43	500 100 2	relu relu softmax	mean_squared_error	Adamax	50	35	0.63	0.628
44	500 100 2	relu relu softmax	mean_squared_error	Nadam	50	35	0.63	0.628
45	500 100 2	relu relu softmax	squared_hinge	Nadam	50	35	0.37	0.372
46	500 100 2	relu relu softmax	categorical_hinge	Nadam	50	35	0.37	0.372
47	500 100 2	relu relu softmax	logcosh	Nadam	50	35	0.63	0.628
48	500 100 2	sigmoid sigmoid softmax	mean_squared_error	sgd	50	35	0.57	0.572

```
2612/2612 [==============================] - 5s 2ms/step - loss: 0.4224 - accuracy: 0.8041
1120/1120 [==============================] - 2s 2ms/step - loss: 0.4282 - accuracy: 0.8011
Train accuracy:0.804
Test accuracy:0.801
```

模型介紹-MLP

```
from sklearn.neural_network import MLPClassifier
mlp = MLPClassifier(hidden_layer_sizes=(X.shape[1]),
activation='relu',
solver='adam',batch_size='auto',
learning_rate='constant',
learning_rate_init=0.001,
max_iter=200,
random_state=0)

#  mlp.summary()
```

```
mlp.fit(X_train,y_train)
print("Train accuracy of MLP:{:.3f}".format(mlp.score(X_train,y_train)))
print("Test accuracy of MLP:{:.3f}".format(mlp.score(X_test,y_test)))
```

```
Train accuracy of MLP:0.824
Test accuracy of MLP:0.819
```

```
from sklearn.metrics import f1_score
predict = mlp.predict(X_test)
print(f1_score(predict, y_test))
```

```
0.7230880908663905
```

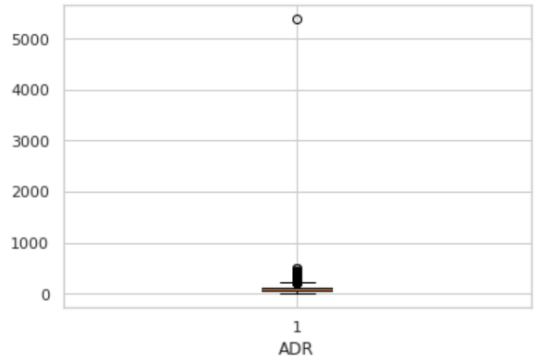
模型介紹-小結

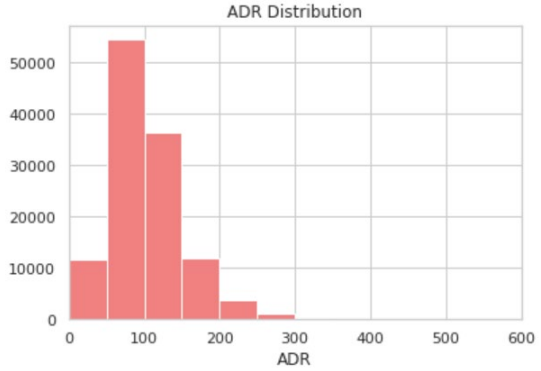
	RandomForest	NN+	MLP
學習程度	Machine Learning	Deep Learning	Deep Learning
套件	sklearn	keras	sklearn
調整空間	少	多	多
調整速度	快	慢	中
準確度	86.64%	80.4%	81.9%

-資料優化-

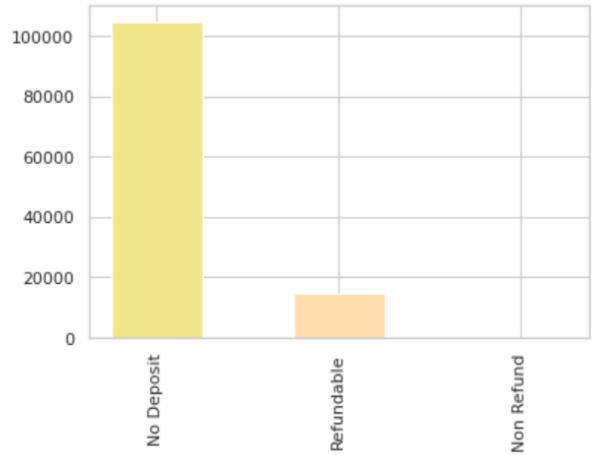
資料優化-離群值處理

ADR

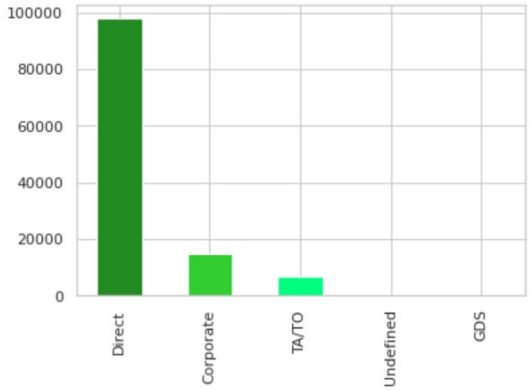




Deposit Type



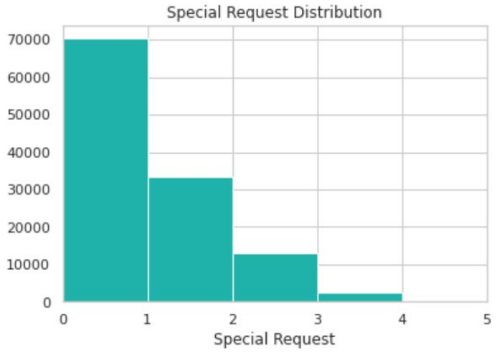
Distribution Channel




```
data = data[(data['distribution_channel'] == 'TA/TO') |
            (data['distribution_channel'] == 'Direct') |
            (data['distribution_channel'] == 'Corporate')]
```

資料優化-資料重分類

Special Request

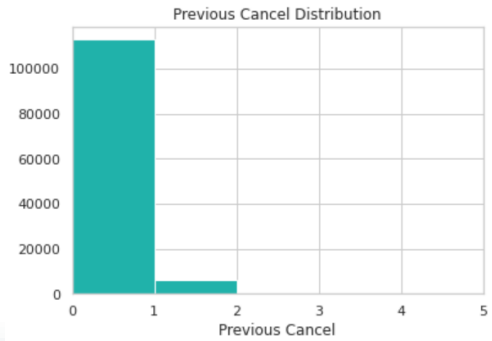


Booking Changes

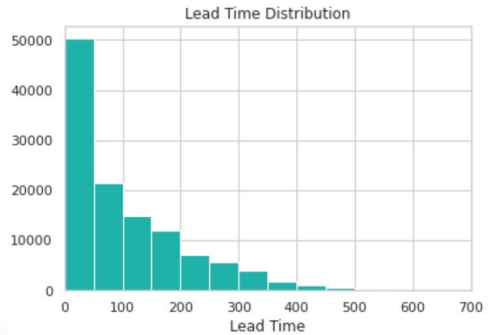
Parking Space


```
data['changed_booking']  =  pd.Series([0  if  x  ==  0  else  1  for  x  in  data['booking_changes']])
data['changed_booking']  =  pd.Categorical(data['changed_booking'])
```

Previous Cancel



Lead time



資料優化-NN+

```
#NN+
from  keras.layers  import  Dropout
from  keras.models  import  Sequential
from  keras.layers  import  Dense
model=Sequential()
model.add(Dense(500,input_dim=X.shape[1],activation='relu'))
model.add(Dropout(0.25))
model.add(Dense(100,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])

model.fit(X_train,y_train,epochs=50,batch_size=35,validation_data=(X_test,y_test))
```

```
2480/2480 [==============================] - 3s 1ms/step - loss: 0.2808 - accuracy: 0.8690
1063/1063 [==============================] - 1s 1ms/step - loss: 0.3031 - accuracy: 0.8618
nTrain accuracy:0.869
nTest accuracy:0.862
```

資料優化-NN+

dense	500,relu
dropout	0.25
dense	100,tanh
dropout	0.2
dense	1,softmax
loss_function	mean_squared_error
optimizer	sgd
epochs	constant
batch_size	constant
train_accuracy	0.63
test_accuracy	0.628

dense	500,relu
dropout	0.25
dense	300,tanh
dropout	0.2
dense	1,softmax
loss_function	mean_squared_error
optimizer	sgd
epochs	constant
batch_size	constant
train_accuracy	0.63
test_accuracy	0.628

dense	500,relu
dropout	0.25
dense	100,relu
dropout	0.2
dense	1,softmax
loss_function	mean_squared_error
optimizer	sgd
epochs	constant
batch_size	constant
train_accuracy	0.63
test_accuracy	0.628

dense	500,relu
dropout	0.25
dense	100,relu
dropout	0.2
dense	1,sigmoid
loss_function	mean_squared_error
optimizer	sgd
epochs	constant
batch_size	constant
train_accuracy	0.63
test_accuracy	0.628

dense	500,relu
dropout	0.25
dense	100,relu
dropout	0.2
dense	1,sigmoid
loss_function	binary_crossentropy
optimizer	sgd
epochs	constant
batch_size	constant
train_accuracy	0.63
test_accuracy	0.628

dense	500,relu
dropout	0.25
dense	100,relu
dropout	0.2
dense	1,sigmoid
loss_function	binary_crossentropy
optimizer	adam
epochs	constant
batch_size	constant
train_accuracy	0.869
test_accuracy	0.862

資料優化-MLP

```
1 from sklearn.neural_network import MLPClassifier
2 mlp = MLPClassifier(hidden_layer_sizes=(X.shape[1],40),
3 activation='logistic',
4 solver='adam',batch_size='auto',
5 learning_rate='constant',
6 learning_rate_init=0.001,
7 max_iter=1000,
8 random_state=0)
9
```

```
1 mlp.fit(X_train,y_train)
2 print("Train accuracy of MLP:{:.3f}".format(mlp.score(X_train,y_train)))
3 print("Test accuracy of MLP:{:.3f}".format(mlp.score(X_test,y_test)))
```

```
Train accuracy of MLP:0.892
Test accuracy of MLP:0.873
```

```
1 from sklearn.metrics import f1_score
2 predict = mlp.predict(X_test)
3 print(f1_score(predict, y_test))
```

```
0.8233314177817914
```

資料優化-MLP

hidden_layer_size	X.shape[1]
activation	relu
solver	adam
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	200
train_accuracy	0.875
test_accuracy	0.865
F1_score	0.8098

hidden_layer_size	X.shape[1]
activation	relu
solver	sgd
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	200
train_accuracy	0.836
test_accuracy	0.836
F1_score	0.7663

hidden_layer_size	X.shape[1]
activation	tanh
solver	adam
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	200
train_accuracy	0.879
test_accuracy	0.862
F1_score	0.8164

hidden_layer_size	X.shape[1]
activation	logistic
solver	adam
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	200
train_accuracy	0.885
test_accuracy	0.868
F1_score	0.821

hidden_layer_size	X.shape[1]
activation	logistic
solver	adam
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	1000
train_accuracy	0.888
test_accuracy	0.87
F1_score	0.8219

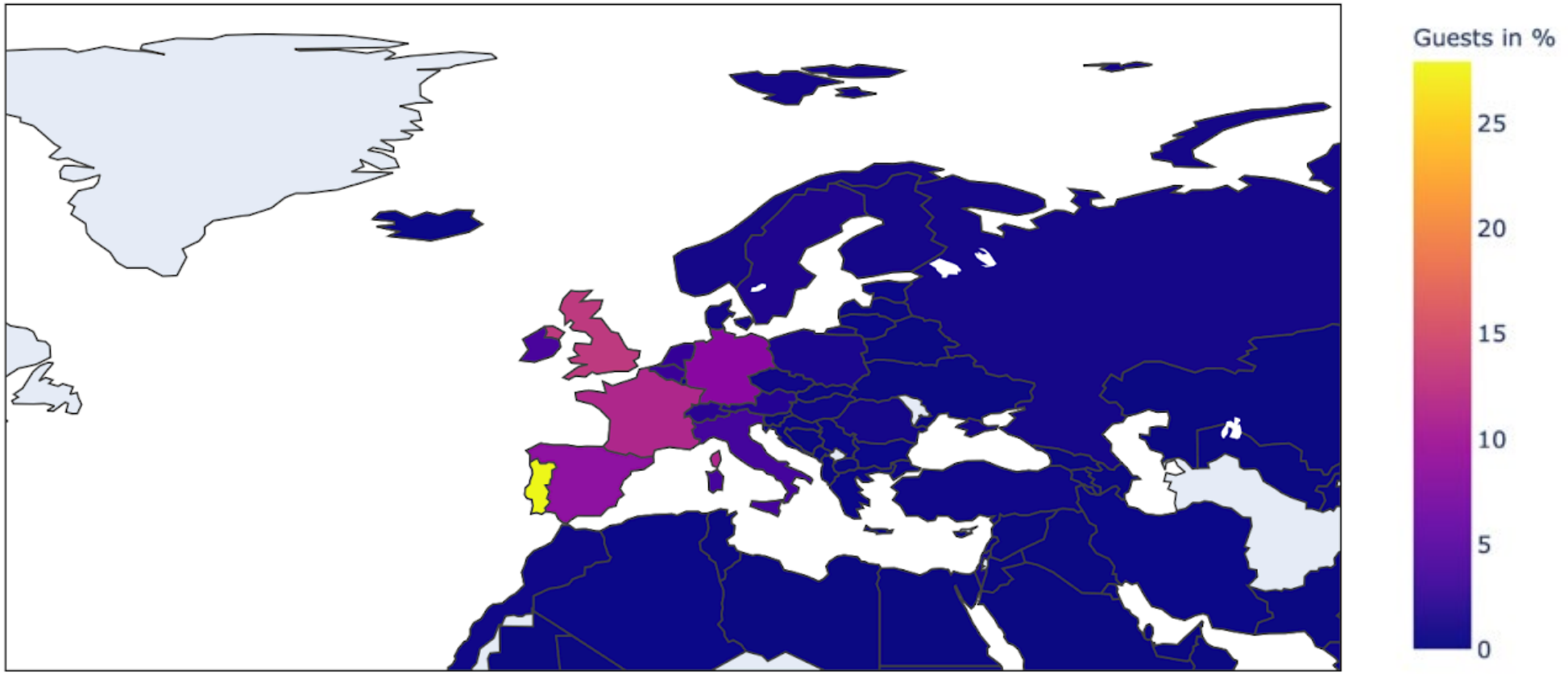
hidden_layer_size	X.shape[1],40
activation	logistic
solver	adam
batch_size	auto
learning_rate	constant
learning_rate_init	0.001
max_iter	1000
train_accuracy	0.892
test_accuracy	0.873
F1_score	0.8233

資料優化- 模型比較

模型	Training Process	資料前處理	資料優化	Optimal Model	Improvement
NN+	• activation • optimizer • loss_function • epochs • batch_size	0.801	0.862	0.862	+6.1%
MLP	• activation • solver • max_iter • hidden_layer_sizes	0.819	0.865	0.873	+5.4%

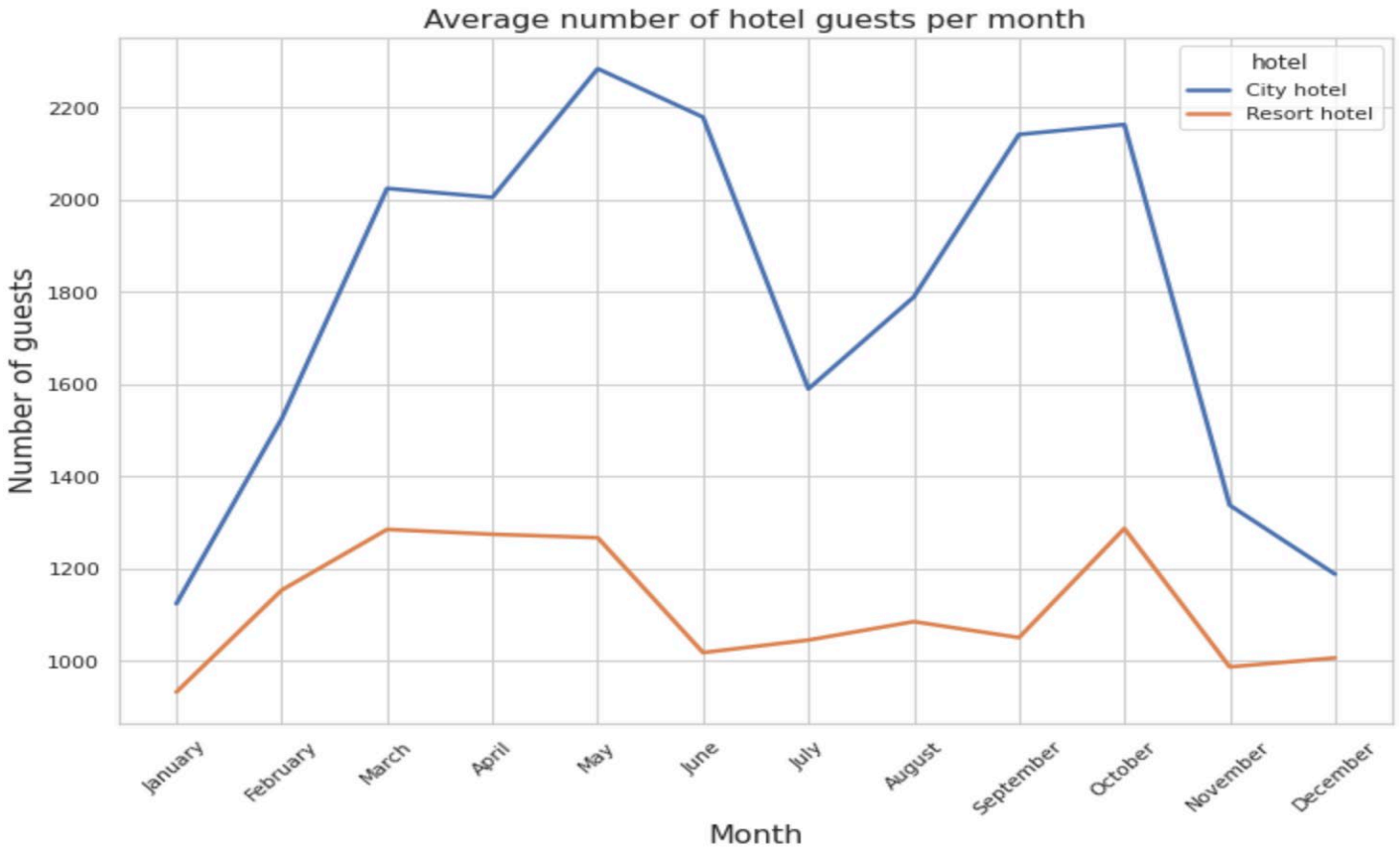
資料優化- 結論

住房地區分析



資料優化-結論

平均每月住客數



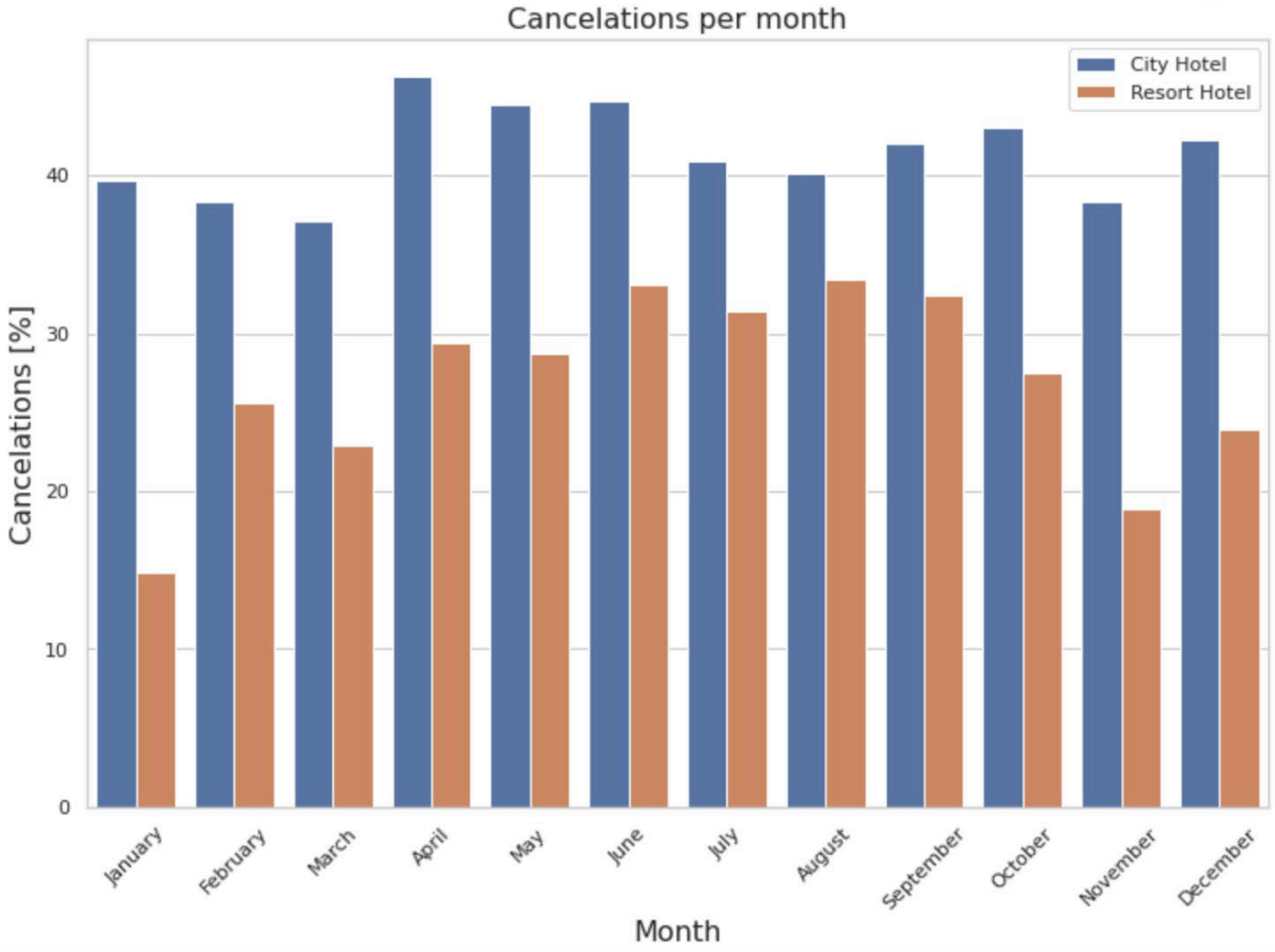
資料優化- 結論

每月住房取消比率

預訂取消總數
44,199 (37 %)

渡假飯店取消總數
11,120 (28 %)

商務飯店取消總數
33,079 (42 %)




Q & A