CHECK FOR NULL AND DUPLICATE

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36285 entries, 0 to 36284
Data columns (total 17 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Booking_ID              36285 non-null  object
 1   number of adults        36285 non-null  int64
 2   number of children      36285 non-null  int64
 3   number of weekend nights 36285 non-null  int64
 4   number of week nights   36285 non-null  int64
 5   type of meal            36285 non-null  object
 6   car parking space       36285 non-null  int64
 7   room type               36285 non-null  object
 8   lead time               36285 non-null  int64
 9   market segment type     36285 non-null  object
 10  repeated                36285 non-null  int64
 11  P-C                     36285 non-null  int64
 12  P-not-C                 36285 non-null  int64
 13  average price           36285 non-null  float64
 14  special requests        36285 non-null  int64
 15  date of reservation     36285 non-null  object
 16  booking status          36285 non-null  object
dtypes: float64(1), int64(10), object(6)
memory usage: 4.7+ MB
```

```
data.duplicated().value_counts()
```

✓ 0.0s

```
False    36285
Name: count, dtype: int64
```

- As we see ther is no duplicates or null values

marwan abbas mohamed

EXTRACTING FEATURES

```python
def month_arr(date):
    parts = date.split('/')
    if len(parts) == 1:
        parts = date.split('-')
        mon = parts[1]
        day = parts[2]
    else:
        mon = parts[0]
        day = parts[1]
    return mon,day
```

✓ 0.0s

```python
def more_than_year(data):
    year = 0
    while data >12 :
        data = data-12
        year += 1
    return data
```

✓ 0.0s

```python
data[['month', 'day']] = pd.DataFrame(data['date of reservation'].apply(month_arr).tolist(), index=data.index)
data['month'] = data['month'].astype(int)

data['day'] = data['day'].astype(int)
data['month of arrive'] = (data['month'] + (data['day'] + data['lead time'])//30)

data['month of arrive'] = pd.DataFrame(data['month of arrive'].apply(more_than_year).tolist(), index=data.index
```

✓ 0.0s

- extractied more relvent feauter date of check in

FEATUER ENCODING

```
data = pd.get_dummies(data,columns=["type of meal" , "room type","market segment type" ],dtype='int8')
data['booking status'] = data['booking status'] != 'Canceled'
data['booking status'] = data['booking status'].astype('int8')
```

✓ 0.0s

- Used one-hot encoding for categorical features that have more than two values and dosnt have proper order.

FEATURE SELECTION

RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier
import pandas as pd

model = RandomForestClassifier()
model.fit(X_encoded, y_encoded)

importances = model.feature_importances_
features = X_encoded.columns

feature_importance = pd.DataFrame({
    'Feature': features,
    'Importance': importances
})

feature_importance.sort_values(by='Importance', ascending=False, inplace=True)

print(feature_importance)
```

	Feature	Importance
5	lead time	0.368756
7	average price	0.196177
8	special requests	0.105632
9	reservation_month	0.097990
3	number of week nights	0.060241
2	number of weekend nights	0.041776
22	market segment type_Online	0.026060
0	number of adults	0.024146
21	market segment type_Offline	0.016221
10	type of meal_Meal Plan 2	0.011862
12	type of meal_Not Selected	0.009650
15	room type_Room_Type 4	0.009454
1	number of children	0.007527
4	car parking space	0.007008
20	market segment type_Corporate	0.006358
6	repeated	0.003250
13	room type_Room_Type 2	0.002726
17	room type_Room_Type 6	0.001971
16	room type_Room_Type 5	0.001819
19	market segment type_Complementary	0.000797
18	room type_Room_Type 7	0.000483
11	type of meal_Meal Plan 3	0.000054
14	room type_Room_Type 3	0.000043

FEATURE SELECTION

Use PCA to reduced dimensions to only 9 components.

```
Final shape after RF + PCA: (28998, 9)
```

FEATURE SELECTION

Checking multicollinearity

	Feature	VIF
0	number of adults	1.34
1	number of children	1.80
2	number of weekend nights	1.07
3	number of week nights	1.10
4	car parking space	1.03
5	lead time	1.20
6	repeated	1.77
7	P-C	1.35
8	P-not-C	1.61
9	average price	1.84
10	special requests	1.23
11	month of arrive	1.06
12	type of meal_Meal Plan 1	inf
13	type of meal_Meal Plan 2	inf
14	type of meal_Meal Plan 3	inf
15	type of meal_Not Selected	inf
16	room type_Room_Type 1	inf
17	room type_Room_Type 2	inf
18	room type_Room_Type 3	inf
19	room type_Room_Type 4	inf
20	room type_Room_Type 5	inf
21	room type_Room_Type 6	inf
22	room type_Room_Type 7	inf
23	market segment type_Aviation	inf
24	market segment type_Complementary	inf
25	market segment type_Corporate	inf
26	market segment type_Offline	inf
27	market segment type_Online	inf

VIF > 10 VERY STRONG MULTICOLLINEARITY

5 < VIF <10 STRONG MULTICOLLINEARITY

VIF <5 ACCEPTABLE RANGE

fisher exact test

```
P-value of number of adults:  0.0001
P-value of number of children:  0.0001
P-value of number of weekend nights:  0.0001
P-value of number of week nights:  0.0001
P-value of car parking space:  8.418005015790401e-47
P-value of lead time:  0.9784
P-value of repeated:  2.259552576228504e-106
P-value of P-C:  0.0001
P-value of P-not-C:  0.0001
P-value of average price :  1.0
P-value of special requests:  0.0001
P-value of month of arrive:  0.0001
P-value of type of meal_Meal Plan 1:  8.345191798624644e-10
P-value of type of meal_Meal Plan 2:  2.3727305250857149e-07
P-value of type of meal_Meal Plan 3:  1.0
P-value of type of meal_Not Selected:  0.0008080292683968222
P-value of room type_Room_Type 1:  2.988800770203236e-11
P-value of room type_Room_Type 2:  0.7075912466584158
P-value of room type_Room_Type 3:  0.6740157416309502
P-value of room type_Room_Type 4:  3.0649223041411686e-09
P-value of room type_Room_Type 5:  0.1551454422220999
P-value of room type_Room_Type 6:  4.369664283097506e-07
P-value of room type_Room_Type 7:  0.013887477752716135
P-value of market segment type_Aviation:  0.8303752998246963
P-value of market segment type_Complementary:  2.318068886348816e-51
P-value of market segment type_Corporate:  2.4616513788841826e-84
P-value of market segment type_Offline:  2.2457680048682087e-38
P-value of market segment type_Online:  1.0895799939563126e-126
```

P-VALUE <0.05 ACCEPTABLE

chi2 test

	features	chi2
0	number of adults	27.758052
1	number of children	49.135815
2	number of weekend nights	161.481095
3	number of week nights	243.848099
4	car parking space	166.839467
5	lead time	324588.422812
6	repeated	311.366825
7	P-C	190.817668
8	P-not-C	1983.948274
9	average price	6412.783619
10	special requests	1655.427638
11	month of arrive	718.134650
12	type of meal_Meal Plan 1	8.545280
13	type of meal_Meal Plan 2	25.310415
14	type of meal_Meal Plan 3	0.060135
15	type of meal_Not Selected	9.660096
16	room type_Room_Type 1	10.107232
17	room type_Room_Type 2	0.134044
18	room type_Room_Type 3	0.552157
19	room type_Room_Type 4	29.536233
20	room type_Room_Type 5	2.212172
21	room type_Room_Type 6	26.333171
22	room type_Room_Type 7	6.214812
23	market segment type_Aviation	0.073938
24	market segment type_Complementary	140.564017
25	market segment type_Corporate	299.963153
26	market segment type_Offline	118.944465
27	market segment type_Online	193.242402

CHI2 > 5

THIS TEST WORK FOR CATEGRIAL FEATUERS

FEATURE SELECTION

```python
data.drop(["Booking_ID","date of reservation","day","month"],axis=1,inplace=True)
```
✓ 0.0s

```python
X = data.drop(["booking status",'market segment type_Offline','type of meal_Not Selected'
              ,"type of meal_Meal Plan 3",'room type_Room_Type 2','room type_Room_Type 3'
              ,'room type_Room_Type 5','market segment type_Aviation'],axis=1)
```

- Removed unnecessary features after checking their relevance.
- Checked VIF after removing collinear features

	Feature	VIF
0	number of adults	1.32
1	number of children	1.77
2	number of weekend nights	1.07
3	number of week nights	1.10
4	car parking space	1.03
5	lead time	1.19
6	repeated	1.76
7	P-C	1.35
8	P-not-C	1.61
9	average price	1.80
10	special requests	1.23
11	month of arrive	1.06
12	type of meal_Meal Plan 1	1.70
13	type of meal_Meal Plan 2	1.74
14	room type_Room_Type 1	7.15
15	room type_Room_Type 4	6.81
16	room type_Room_Type 6	2.26
17	room type_Room_Type 7	1.16
18	market segment type_Complementary	1.32
19	market segment type_Corporate	1.52
20	market segment type_Online	1.76

chi2 for contnius feature after binning to test it

20	lead_time_bin	3075.569879
21	avg_bin	572.168737

OUTLIERS , SPLITE AND SCALING

```
data = remove_outliers_iqr(data, 'lead time')
data = remove_outliers_zscore(data, 'average price ')
```

- **Used IQR for outlier detection in lead time, and Z-score for average price because it follows a normal distribution**

```
X_train,x_test,y_train,y_test = train_test_split(X,Y ,train_size=0.75, test_size=0.25 ,random_state=42)
✓ 0.0s
```

- splite data to train and test

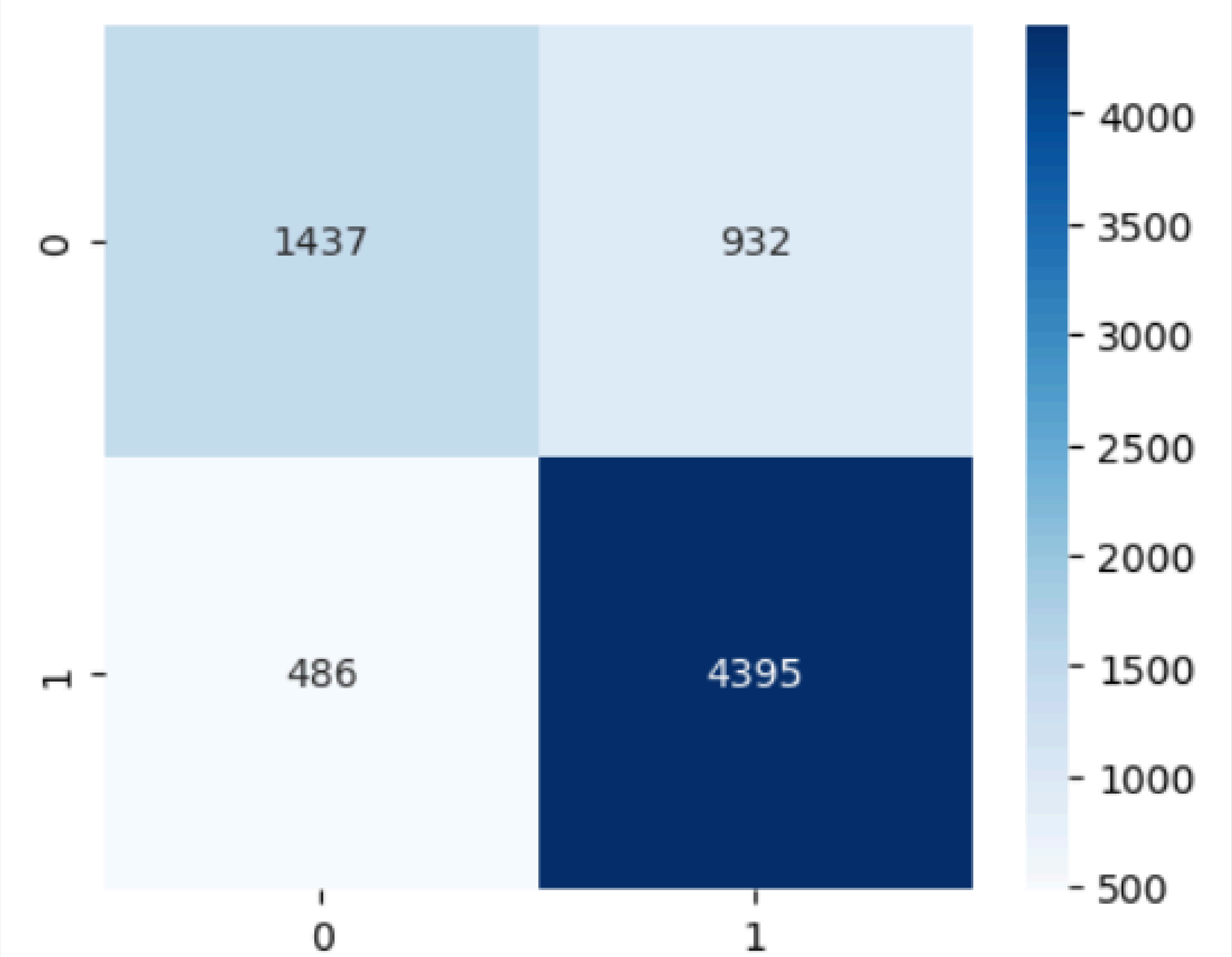
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(x_test)
✓ 0.0s
```

- Applied standard scaling to normalize the data and improve model performance.

MODELING AND ACCURACY CALCULATION

- First use SVC Modeling.

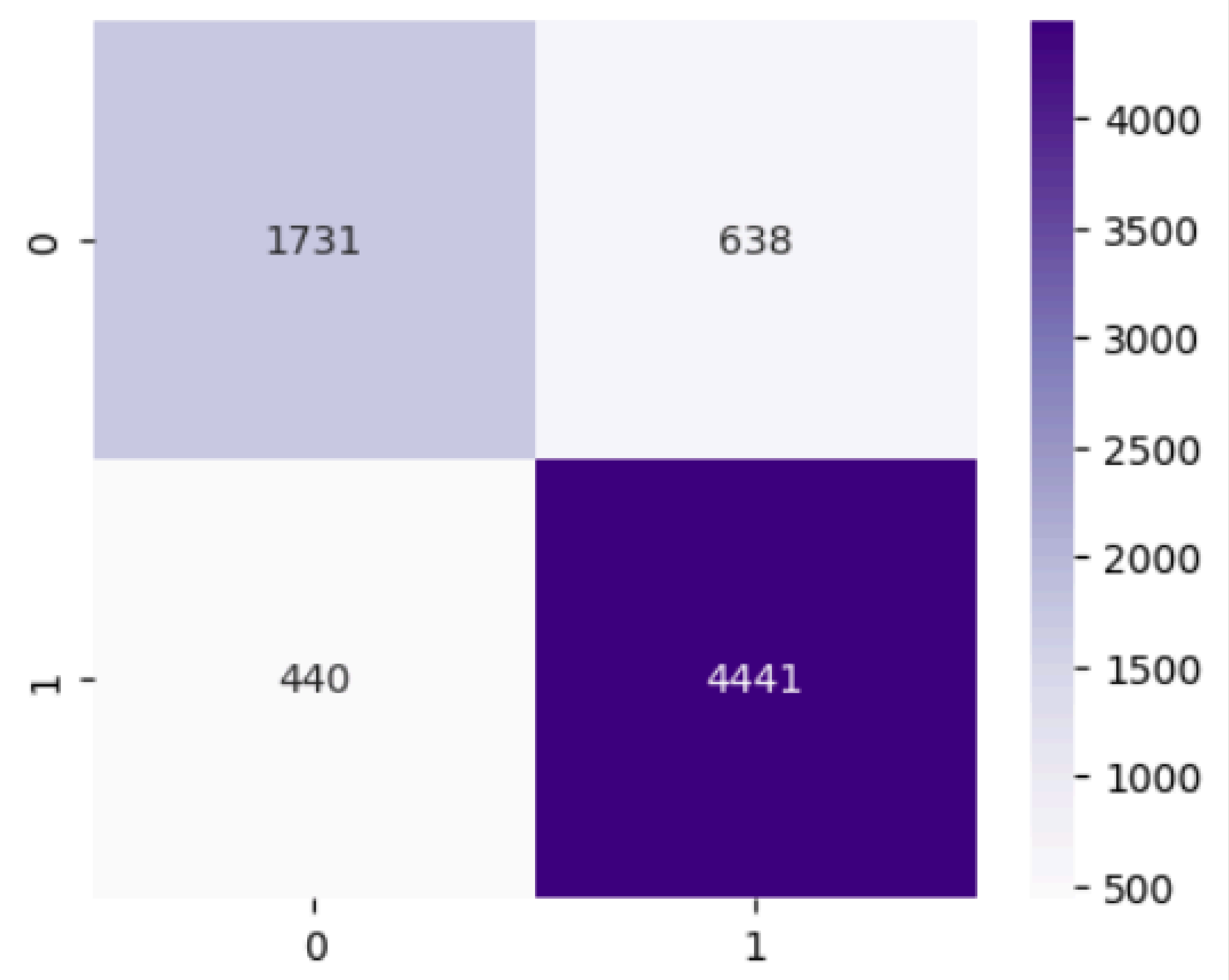
	precision	recall	f1-score	support
0	0.75	0.61	0.67	2369
1	0.83	0.90	0.86	4881
accuracy			0.80	7250
macro avg	0.79	0.75	0.77	7250
weighted avg	0.80	0.80	0.80	7250



MODELING AND ACCURACY CALCULATION

- xgboost Model.

	precision	recall	f1-score	support
0	0.80	0.73	0.76	2369
1	0.87	0.91	0.89	4881
accuracy			0.85	7250
macro avg	0.84	0.82	0.83	7250
weighted avg	0.85	0.85	0.85	7250



MODELING AND ACCURACY CALCULATION

- RandomForestCassifier

	precision	recall	f1-score	support
0	0.79	0.75	0.77	2369
1	0.88	0.90	0.89	4881
accuracy			0.85	7250
macro avg	0.84	0.83	0.83	7250
weighted avg	0.85	0.85	0.85	7250