This analysis is to find out:

1. how do we promote Jewel attractions to rest of population and;
2. how to get Jewel attraction go-ers to spend even more with us

For this, we looked at:

1. what is the propotion of Jewel's attraction visitors spent in the airport before or after visiting the attractions
2. What is the average complimentary spending?
3. What do they spent on?
4. Days between purchase of attraction pass/ticket and the actual visits
5. Time of visits

In [ ]:
```
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.pyplot as plt
import seaborn as sns

pd.options.display.max_colwidth=100
```

In [ ]:
```
jewel_txn = pd.read_csv('JCASG-20210702-MemberTransaction.csv')

jewel_cust_id = pd.read_csv('jewel_customer id.csv')
jewel_cust_id.columns = ['HashEmail', 'CUSTOMERID', 'CR_Tier']

txn_data = pd.read_csv('txn_Oct20 to June21.csv')
```

In [ ]:
```
jewel_txn['TransactionDate'] = pd.to_datetime(jewel_txn['TransactionDate'])
jewel_txn['MembershipStartDate'] = pd.to_datetime(jewel_txn['MembershipStartDate'], format='%Y-%m-%d'
txn_data['TRANSACTIONDATE'] = pd.to_datetime(txn_data['TRANSACTIONDATE'])

jewel_txn['MembershipExpiryDate'] = np.where(jewel_txn['MembershipExpiryDate']=='9999-12-31',
                                             '2099-12-31',
                                             jewel_txn['MembershipExpiryDate'])
jewel_txn['MembershipExpiryDate'] = pd.to_datetime(jewel_txn['MembershipExpiryDate'], format='%Y-%m-%
```

In [ ]:
```
jewel_txn['membership_flag'] = np.where(jewel_txn['MembershipType']=='Jewel Shopper',
                                        1,
                                        0)
jewel_txn['HashEmail'] = jewel_txn['HashEmail'].str.upper()
```

In [ ]:
```
jewel_txn = jewel_txn.merge(right=jewel_cust_id, on='HashEmail', how='left').drop_duplicates()
```

In [ ]:
```
jewel_txn.head()
```

In [ ]:
```
txn_data.head()
```

In [ ]:
```
JEWEL_ATTRACTIONS =['Attractions', 'Changi Experience Studio']

len(jewel_txn[(jewel_txn['Location'].isin(JEWEL_ATTRACTIONS)) &
              (jewel_txn['TransactionDate']>='2020-10-01') &
              (jewel_txn['TransactionDate']<='2021-03-31')])
```

In [ ]:
```
jewel_txn[(jewel_txn['Location'].isin(JEWEL_ATTRACTIONS)) &
          (jewel_txn['TransactionDate']>='2020-10-01') &
          (jewel_txn['TransactionDate']<='2021-03-31')&
          ~(jewel_txn['CUSTOMERID'].isna())].CUSTOMERID.nunique()
```

In [ ]:
```
jewel_txn['TransactionMonth'] = jewel_txn['TransactionDate'].dt.strftime('%Y-%m')

txn_attr_only = jewel_txn[(jewel_txn['Location'].isin(JEWEL_ATTRACTIONS)) & ~(jewel_txn['CUSTOMERID']
to_plot = txn_attr_only.groupby('TransactionMonth').agg({'CUSTOMERID':np.count_nonzero}).reset_index(

fig, ax = plt.subplots(figsize=(15,5))
sns.barplot(x='TransactionMonth', y='CUSTOMERID', data=to_plot,
            ax=ax)
```

1 Complimentary spending before or after visiting Jewel's attractions

In [ ]:

```
jewel_txn['unique_mdm_date'] = (jewel_txn['CUSTOMERID']
                                + '_'
                                + jewel_txn['TransactionDate'].dt.strftime('%Y-%m-%d'))

txn_data['unique_mdm_date'] = (txn_data['CUSTOMERID']
                               + '_'
                               + txn_data['TRANSACTIONDATE'].dt.strftime('%Y-%m-%d'))
```

In [ ]:

```
txn_data.head()
```

In [ ]:

```
member_visited_attr = jewel_txn[
    ~(jewel_txn['CUSTOMERID'].isna())
    &
    (jewel_txn['Location'].isin(JEWEL_ATTRACTIONS))
    ].drop_duplicates(['unique_mdm_date'])
lst_member_visited_attr = member_visited_attr['unique_mdm_date'].to_list()
```

In [ ]:

```
txn_compliment_attr = txn_data[txn_data['unique_mdm_date'].isin(lst_member_visited_attr)]
```

In [ ]:

```
w_other_spendings = txn_compliment_attr.CUSTOMERID.nunique()
spent_at_attr = (jewel_txn[(jewel_txn['Location'].isin(JEWEL_ATTRACTIONS)) &
                           ~(jewel_txn['CUSTOMERID'].isna())].CUSTOMERID.nunique())

(w_other_spendings/spent_at_attr) * 100
```

In [ ]:

```
w_other_spendings
```

In [ ]:

```
len(member_visited_attr)
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(10, 5), subplot_kw=dict(aspect="equal"))

labels = ["{} visitors spent on other products/service on day of visit".format(w_other_spendings),
          ""]

data = [w_other_spendings, spent_at_attr-w_other_spendings]

wedges, texts = ax.pie(data, wedgeprops=dict(width=0.5), startangle=-40)

bbox_props = dict(boxstyle="square,pad=0.3", fc="w", ec="k", lw=0.72)
kw = dict(arrowprops=dict(arrowstyle="-"),
          bbox=bbox_props, zorder=0, va="center")

for i, p in enumerate(wedges):
    ang = (p.theta2 - p.theta1)/2. + p.theta1
    y = np.sin(np.deg2rad(ang))
    x = np.cos(np.deg2rad(ang))
    horizontalalignment = {-1: "right", 1: "left"}[int(np.sign(x))]
    connectionstyle = "angle,angleA=0,angleB={}".format(ang)
    kw["arrowprops"].update({"connectionstyle": connectionstyle})
    ax.annotate(labels[i], xy=(x, y), xytext=(1.35*np.sign(x), 1.4*y),
                horizontalalignment=horizontalalignment, **kw)

ax.set_title("{} visited Jewel's attractions".format(spent_at_attr))

plt.show()
```

In [ ]:

```
to_plot = txn_compliment_attr.groupby(['PRODNM']).agg(
    {
        'NETSPEND':[np.count_nonzero, np.sum]
    }).droplevel(0, axis=1).sort_values('sum', ascending=False).reset_index()
```

In [ ]:

```
to_plot
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(15,5))
ax.set_xticklabels(to_plot['PRODNM'], rotation=50, ha='right')

sns.barplot(x='PRODNM', y='sum', data=to_plot,
            ax=ax)
```

1.1 in Jewel or Terminal??

In [ ]:

```
txn_compliment_attr['jewel_flag'] = np.where(txn_compliment_attr['LOCATIONNM']=='CAG JEWEL',
                                             'Jewel',
                                             'Terminals')
```

In [ ]:

```
txn_compliment_attr.head()
```

In [ ]:

```
txn_compliment_attr.groupby('jewel_flag').agg({'NETSPEND':np.sum})
```

2 Do the attractions encourage more visits and spendings?

In [ ]:

```
txn_data.head(1)
```

2.1 visits

In [ ]:

```
member_visited_attr = txn_compliment_attr.CUSTOMERID.drop_duplicates().to_list()
member_visited_attr_today = txn_compliment_attr.unique_mdm_date.drop_duplicates().to_list()
```

In [ ]:

```
txn_data_21 = txn_data.copy()
```

In [ ]:

```
txn_data_21['attr_flag'] = np.where(txn_data_21['CUSTOMERID'].isin(member_visited_attr),
                                    'visited_attr',
                                    'x_visited_attr')
```

In [ ]:

```
txn_data_21.drop_duplicates(['unique_mdm_date'], inplace=True)
```

In [ ]:

```
cust_grouped = txn_data_21.groupby(['CUSTOMERID', 'attr_flag']).agg({'TRANSACTIONDATE':np.count_nonze
```

In [ ]:

```
cust_grouped.head()
```

In [ ]:

```
cust_grouped.groupby(['attr_flag']).agg({'TRANSACTIONDATE':[np.mean]}).droplevel(0, axis=1)
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(15,7))
ax.set_xticks([x+0.5 for x in range(0,11)])
ax.set_xticklabels([x for x in range(0,11)])

ax.hist(x=cust_grouped[cust_grouped['attr_flag']=='visited_attr'].TRANSACTIONDATE,
        bins=[x for x in range(0,11)],
        density=True)

ax.hist(x=cust_grouped[cust_grouped['attr_flag']=='x_visited_attr'].TRANSACTIONDATE,
        bins=[x for x in range(0,11)],
        density=True)

ax.set_xlabel('number of visits to Airport')
ax.set_ylabel('percentage of population')
ax.legend(["visited Jewel's attractions", "never visited Jewel's attractions"])
```

2.2 average spendings

In [ ]:

```
cust_grouped_2 = txn_data_21.groupby(
    ['CUSTOMERID', 'attr_flag']).agg({'NETSPEND':np.sum}).reset_index()
```

In [ ]:

```
cut_off_pct = np.percentile(cust_grouped_2['NETSPEND'], 99)

cust_grouped_2[cust_grouped_2['NETSPEND']<=cut_off_pct].groupby(['attr_flag']).agg({'NETSPEND':[np.me
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(15,5))
ax.set_xticks([x for x in range(0,1600,100)])
#ax.set_xticklabels([x for x in range(0,1000,100)])

ax.hist(x=cust_grouped_2[cust_grouped_2['attr_flag']=='visited_attr'].NETSPEND,
        bins=[x for x in range(0,1600,100)],
        density=True)m

ax.hist(x=cust_grouped_2[cust_grouped_2['attr_flag']=='x_visited_attr'].NETSPEND,
        bins=[x for x in range(0,1600,100)],
        density=True)
```

2.3 Spending category

In [ ]:

```
cust_grouped_4 = txn_data_21.groupby(['attr_flag', 'PRODNM']).agg(
    {'NETSPEND':np.sum})
cust_grouped_4.unstack(0).droplevel(0, axis=1).fillna(0).sort_values('visited_attr', ascending=False)
```

In [ ]:

```
LUXURIES =['Watches', 'Luxury Brand Name', 'Jewellery']

cust_grouped_4 = txn_data_21[txn_data_21['PRODNM'].isin(LUXURIES)].groupby(['attr_flag', 'PRODNM']).a
    {'NETSPEND':np.mean})
cust_grouped_4.unstack(0).droplevel(0, axis=1).fillna(0).sort_values('x_visited_attr', ascending=Fals
```

In [ ]:

```
FOOD =['Specialty Restaurant', 'Café', 'Fast Food']

cust_grouped_4 = txn_data_21[txn_data_21['PRODNM'].isin(FOOD)].groupby(['attr_flag', 'PRODNM']).agg(
    {'NETSPEND':np.mean})
cust_grouped_4.unstack(0).droplevel(0, axis=1).fillna(0).sort_values('x_visited_attr', ascending=Fals
```

In [ ]:

```
len(txn_data_21[txn_data_21['PRODNM']=='Watches'])
```

3 Membership details

In [ ]:

```
txn_compliment_attr.head()
```

In [ ]:

```
jewel_txn_unique_mdm = jewel_txn.drop_duplicates(['CUSTOMERID'])
```

In [ ]:

```
txn_compliment_attr_expanded = txn_compliment_attr.merge(
    right=jewel_txn_unique_mdm[['CUSTOMERID', 'MembershipType', 'CR_Tier']],
    on='CUSTOMERID',
    how='left').drop_duplicates()
```

In [ ]:

```
txn_compliment_attr_expanded['pass_flag'] = np.where(txn_compliment_attr_expanded['MembershipType']==
                                                     'x_pass_holder',
                                                     'pass_holder')
```

In [ ]:

```
txn_compliment_attr.head()
```

3.1 Spending difference

In [ ]:

```
txn_compliment_attr_expanded.groupby(['pass_flag']).agg({'NETSPEND':np.mean})
```

3.2 CR memberships

In [ ]:

```
visited_attr = jewel_txn[(jewel_txn['Location'].isin(JEWEL_ATTRACTIONS))].drop_duplicates(['CUSTOMERI
visited_attr.CR_Tier.fillna('tba', inplace=True)
```

In [ ]:
```
visited_attr.head()
```

In [ ]:
```
visited_attr.groupby(['CR_Tier']).agg(
    {'CUSTOMERID':np.count_nonzero}).sort_values('CUSTOMERID', ascending=False)
```

In [ ]:
```
visited_attr.CUSTOMERID.nunique()
```

In [ ]:
```
visited_attr.head()
```

4 Days between pruchase of memberships and visits

In [ ]:
```
jewel_txn_34 = jewel_txn.copy()
```

In [ ]:
```
jewel_txn_34.head()
```

In [ ]:
```
JEWEL_ATTRACTIONS =['Attractions', 'Changi Experience Studio']

jewel_txn_34 = jewel_txn_34[jewel_txn_34['Location'].isin(JEWEL_ATTRACTIONS)]
jewel_txn_34.sort_values(['TransactionDate'], inplace=True)
jewel_txn_34.drop_duplicates(['HashEmail'], inplace=True)
```

In [ ]:
```
jewel_txn_34['time_diff'] = (jewel_txn_34['TransactionDate'] - jewel_txn_34['MembershipStartDate']).d
#jewel_txn_34['day_diff'] = jewel_txn_34['time_diff'].days
```

In [ ]:
```
jewel_txn_34.sort_values('time_diff', ascending=False).head()
```

In [ ]:
```
fig, ax = plt.subplots(figsize=(15,5))
#ax.set_xticks([x for x in range(0,600,10)])

ax.hist(x=jewel_txn_34[jewel_txn_34['time_diff']>=0]['time_diff'],
        bins=[x for x in range(0,100,1)],
        density=True)
```

In [ ]:
```
len(jewel_txn_34[jewel_txn_34['time_diff']==0]) / len(jewel_txn_34)
```

In [ ]:
```
same_day_txn = jewel_txn_34[jewel_txn_34['time_diff']==0]
```

In [ ]:
```
same_day_txn['time_visited'] = same_day_txn['TransactionDate'].dt.hour
```

In [ ]:
```
to_plot = same_day_txn.groupby(['time_visited']).agg({'CUSTOMERID':np.count_nonzero}).reset_index()

fig,ax = plt.subplots(figsize=[15,5])

plt.bar(x=to_plot['time_visited'], height=to_plot['CUSTOMERID'])
plt.xlim(0,23)
ax.set_xticks([x for x in range(0,24,1)])
ax.set_xticklabels([x for x in range(0,24,1)])
```

In [ ]:
```

```