**Project - 1**

**Travel Aggregator Analysis**

* Kavidharsini Nataraj(2566298)

**1. Find the Number of Distinct Bookings, Sessions, and Searches:**

import pandas as pd

bookings = pd.read\_csv('Bookings.csv')

sessions = pd.read\_csv('Sessions.csv')

distinct\_bookings = bookings['booking\_id'].nunique()

distinct\_sessions = sessions['session\_id'].nunique()

distinct\_searches = sessions['search\_id'].nunique()

print(f'Distinct Bookings: {distinct\_bookings}')

print(f'Distinct Sessions: {distinct\_sessions}')

print(f'Distinct Searches: {distinct\_searches}')

**2. How Many Sessions Have More Than One Booking?**

sessions\_booking\_count = sessions.groupby('session\_id')['booking\_id'].count()

sessions\_with\_multiple\_bookings = sessions\_booking\_count[sessions\_booking\_count > 1].count()

print(f'Sessions with more than one booking: {sessions\_with\_multiple\_bookings}')

**3. Which Days of the Week Have the Highest Number of Bookings?**

import matplotlib.pyplot as plt

bookings['booking\_time'] = pd.to\_datetime(bookings['booking\_time'])

bookings['day\_of\_week'] = bookings['booking\_time'].dt.day\_name()

day\_of\_week\_counts = bookings['day\_of\_week'].value\_counts()

plt.figure(figsize=(8, 6))

day\_of\_week\_counts.plot.pie(autopct='%1.1f%%', startangle=90, colors=plt.cm.Paired.colors)

plt.title('Distribution of Bookings by Day of the Week')

plt.ylabel('')

plt.show()

**4. Total Number of Bookings and Gross Booking Value for Each Service**

service\_summary = bookings.groupby('service\_name').agg(

total\_bookings=('booking\_id', 'count'),

total\_gross\_booking\_value=('INR\_Amount', 'sum')

)

print(service\_summary)

**5. Most Booked Route for Customers with More Than 1 Booking**

customer\_booking\_counts = bookings.groupby('customer\_id').size()

frequent\_customers = customer\_booking\_counts[customer\_booking\_counts > 1].index

frequent\_customer\_bookings = bookings[bookings['customer\_id'].isin(frequent\_customers)]

most\_booked\_route = frequent\_customer\_bookings.groupby(['from\_city', 'to\_city']).size().idxmax()

print(f"Most booked route: {most\_booked\_route}")

**6. Top 3 Departure Cities with Bookings Made Mostly in Advance**

city\_departures = bookings.groupby('from\_city').size()

cities\_with\_min\_departures = city\_departures[city\_departures >= 5].index

filtered\_bookings = bookings[bookings['from\_city'].isin(cities\_with\_min\_departures)]

city\_avg\_days\_to\_departure = filtered\_bookings.groupby('from\_city')['days\_to\_departure'].mean()

top\_3\_cities = city\_avg\_days\_to\_departure.nlargest(3)

print(f'Top 3 departure cities:\n{top\_3\_cities}')

**7. Heatmap of Correlations of Numerical Columns**

import seaborn as sns

numerical\_columns = bookings[['INR\_Amount', 'no\_of\_passengers', 'days\_to\_departure', 'distance\_km']]

correlation\_matrix = numerical\_columns.corr()

plt.figure(figsize=(10, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt='.2f')

plt.title('Correlation Heatmap')

plt.show()

max\_corr\_pair = correlation\_matrix.unstack().idxmax()

print(f'Pair with maximum correlation: {max\_corr\_pair}')

**8. Most Used Device Type for Each Service**

device\_usage = bookings.groupby('service\_name')['device\_type\_used'].agg(lambda x: x.mode().iloc[0])

print(f'Most used device type for each service:\n{device\_usage}')

**9. Trends for Number of Bookings by Device Type (Quarterly)**

bookings['year'] = bookings['booking\_time'].dt.year

bookings['quarter'] = bookings['booking\_time'].dt.to\_period('Q')

booking\_trends = bookings.groupby(['year', 'quarter', 'device\_type\_used']).size().unstack().fillna(0)

booking\_trends.plot(figsize=(12, 8))

plt.title('Quarterly Booking Trends by Device Type')

plt.ylabel('Number of Bookings')

plt.xlabel('Year and Quarter')

plt.legend(title='Device Type')

plt.show()

**10. Booking to Search Ratio Analysis**

bookings['booking\_time'] = pd.to\_datetime(bookings['booking\_time'], errors='coerce')

sessions['search\_time'] = pd.to\_datetime(sessions['search\_time'], errors='coerce')

merged\_sessions = pd.merge(sessions, bookings[['booking\_id']], on='booking\_id', how='left')

session\_summary = merged\_sessions.groupby('session\_id').agg(

total\_searches=('search\_id', 'count'),

total\_bookings=('booking\_id', 'count')

)

session\_summary['oBSR'] = session\_summary['total\_bookings'] / session\_summary['total\_searches'].replace(0, pd.NA)

session\_summary = session\_summary.join(sessions.set\_index('session\_id')['search\_time'])

session\_summary['search\_time'] = pd.to\_datetime(session\_summary['search\_time'])

session\_summary.set\_index('search\_time', inplace=True)

monthly\_obsr = session\_summary.resample('M').mean()['oBSR']

weekly\_obsr = session\_summary.resample('W').mean()['oBSR']

plt.figure(figsize=(12, 6))

plt.plot(monthly\_obsr.index, monthly\_obsr, label='Monthly oBSR', color='b')

plt.plot(weekly\_obsr.index, weekly\_obsr, label='Weekly oBSR', color='r', alpha=0.7)

plt.title('Time Series of oBSR')

plt.xlabel('Date')

plt.ylabel('oBSR')

plt.legend()

plt.grid(True)

plt.show()