```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from scipy.stats import zscore
```

```
df = pd.read_csv("/content/AeroReach Insights.csv",encoding ='unicode_escape')
```

```
df
```

	UserID	Taken_product	Yearly_avg_view_on_travel_page	preferred_device	tota
0	1000001	Yes	307.0	iOS and Android	
1	1000002	No	367.0	iOS	
2	1000003	Yes	277.0	iOS and Android	
3	1000004	No	247.0	iOS	
4	1000005	No	202.0	iOS and Android	
...	...	...	...	...	
11755	1011756	No	279.0	Laptop	
11756	1011757	No	305.0	Tab	
11757	1011758	No	214.0	Tab	
11758	1011759	No	382.0	Laptop	
11759	1011760	No	270.0	Tab	

11760 rows × 17 columns

```
df.columns
```

```
Index(['UserID', 'Taken_product', 'Yearly_avg_view_on_travel_page',
       'preferred_device', 'total_likes_on_outstation_checkin_given',
       'yearly_avg_Outstation_checkins', 'member_in_family',
       'preferred_location_type', 'Yearly_avg_comment_on_travel_page',
       'total_likes_on_outofstation_checkin_received',
       'week_since_last_outstation_checkin', 'following_company_page',
       'montly_avg_comment_on_company_page', 'working_flag',
       'travelling_network_rating', 'Adult_flag',
       'Daily_Avg_mins_spend_on_traveling_page'],
      dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11760 entries, 0 to 11759
Data columns (total 17 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   UserID                                11760 non-null  int64  
 1   Taken_product                         11760 non-null  object 
 2   Yearly_avg_view_on_travel_page        11179 non-null  float64
 3   preferred_device                      11707 non-null  object 
 4   total_likes_on_outstation_checkin_given  11379 non-null  float64
 5   yearly_avg_Outstation_checkins        11685 non-null  object 
 6   member_in_family                      11760 non-null  object 
 7   preferred_location_type               11729 non-null  object 
 8   Yearly_avg_comment_on_travel_page     11554 non-null  float64
 9   total_likes_on_outofstation_checkin_received  11760 non-null  int64  
 10  week_since_last_outstation_checkin    11760 non-null  int64  
 11  following_company_page                11657 non-null  object 
 12  montly_avg_comment_on_company_page    11760 non-null  int64  
 13  working_flag                          11760 non-null  object 
 14  travelling_network_rating             11760 non-null  int64  
 15  Adult_flag                            11760 non-null  int64  
 16  Daily_Avg_mins_spend_on_traveling_page  11760 non-null  int64  
dtypes: float64(3), int64(7), object(7)
memory usage: 1.5+ MB
```

```
df.describe()
```

```
df.isnull().sum()
```

```
df.shape
```

```
(11760, 17)
```

```
df.head()
```

```
pd.isnull(df)
```

```
df[df.isnull().any(axis=1)]
```

```
df['member_in_family'] = df['member_in_family'].replace('Three', 3)
```

```
df['member_in_family']
```

```
num_cols = df.select_dtypes(include=['int64', 'float64']).columns
for col in num_cols:
    df[col] = df[col].fillna(df[col].median())
```

```
num_cols
```

```
Index(['UserID', 'Yearly_avg_view_on_travel_page',
       'total_likes_on_outstation_checkin_given',
       'Yearly_avg_comment_on_travel_page',
       'total_likes_on_outofstation_checkin_received',
       'week_since_last_outstation_checkin',
       'montly_avg_comment_on_company_page', 'travelling_network_rating',
       'Adult_flag', 'Daily_Avg_mins_spend_on_traveling_page'],
      dtype='object')
```

```
cat_cols = df.select_dtypes(include=['object']).columns
for col in cat_cols:
    df[col] = df[col].fillna(df[col].mode()[0])
```

```
cat_cols
```

```
Index(['Taken_product', 'preferred_device', 'yearly_avg_Outstation_checkins',
       'member_in_family', 'preferred_location_type', 'following_company_page',
       'working_flag'],
      dtype='object')
```

```
print("\nNumber of duplicates:", df.duplicated().sum())
```

```
Number of duplicates: 0
```

```
z_scores = zscore(df.select_dtypes(include=['int64', 'float64']))
abs_z_scores = np.abs(z_scores)
filtered_entries = (abs_z_scores < 3).all(axis=1)
df = df[filtered_entries]
```

```
z_scores
```

```
array([[-1.73190353,  0.40077353,  0.73519169, ..., -1.58417824,
        -0.93201439, -0.64137356],
       [-1.73160896,  1.30291384, -1.30057234, ...,  1.19143799,
         0.24198831, -0.42087298],
       [-1.7313144 , -0.05029662,  1.4055344 , ..., -0.65897283,
        -0.93201439, -0.75162385],
       ...,
       [ 1.7313144 , -0.99754395, -1.60355169, ..., -0.65897283,
         0.24198831, -0.2003724 ],
       [ 1.73160896,  1.52844891,  0.54302914, ...,  1.19143799,
        -0.93201439,  0.68162992],
       [ 1.73190353, -0.15554633, -0.43410933, ..., -1.58417824,
        -0.93201439,  0.02012818]])
```

```
print("\nShape after outlier removal:", df.shape)
```

```
Shape after outlier removal: (11487, 17)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11487 entries, 0 to 11759
Data columns (total 17 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   UserID                                11487 non-null  int64
 1   Taken_product                         11487 non-null  object
 2   Yearly_avg_view_on_travel_page        11487 non-null  float64
 3   preferred_device                      11487 non-null  object
 4   total_likes_on_outstation_checkin_given  11487 non-null  float64
 5   yearly_avg_Outstation_checkins        11487 non-null  object
```

```
 5   yearly_avg_Outstation_checkins               11487 non-null  object
 6   member_in_family                             11487 non-null  object
 7   preferred_location_type                      11487 non-null  object
 8   Yearly_avg_comment_on_travel_page            11487 non-null  float64
 9   total_likes_on_outofstation_checkin_received 11487 non-null  int64
 10  week_since_last_outstation_checkin           11487 non-null  int64
 11  following_company_page                       11487 non-null  object
 12  montly_avg_comment_on_company_page           11487 non-null  int64
 13  working_flag                                 11487 non-null  object
 14  travelling_network_rating                    11487 non-null  int64
 15  Adult_flag                                   11487 non-null  int64
 16  Daily_Avg_mins_spend_on_traveling_page       11487 non-null  int64
dtypes: float64(3), int64(7), object(7)
memory usage: 1.6+ MB
```

Start coding or generate with AI.

Exploratory Data Analysis

```
df.columns
```

```
Index(['UserID', 'Taken_product', 'Yearly_avg_view_on_travel_page',
       'preferred_device', 'total_likes_on_outstation_checkin_given',
       'yearly_avg_Outstation_checkins', 'member_in_family',
       'preferred_location_type', 'Yearly_avg_comment_on_travel_page',
       'total_likes_on_outofstation_checkin_received',
       'week_since_last_outstation_checkin', 'following_company_page',
       'montly_avg_comment_on_company_page', 'working_flag',
       'travelling_network_rating', 'Adult_flag',
       'Daily_Avg_mins_spend_on_traveling_page'],
      dtype='object')
```

```
ax = sns.countplot(x='Adult_flag',hue='Taken_product', data = df)

for bars in ax.containers:
    ax.bar_label(bars)
```

```
sns.countplot(x='Taken_product',hue='Adult_flag', data=df)
```

```
num_cols = ['Yearly_avg_view_on_travel_page', 'yearly_avg_Outstation_checkins',
            'Daily_Avg_mins_spend_on_traveling_page', 'Engagement_score', # Changed 'enga
            'social_interaction']

plt.figure(figsize=(15, 10))
for i, col in enumerate(num_cols, 1):
    plt.subplot(3, 2, i)
    sns.histplot(df[col], kde=True, bins=30)
    plt.title(f'Distribution of {col}')
    plt.tight_layout()
plt.show()
```

```
* Most numerical features are right-skewed, indicating most users have lower values wi

* Yearly_avg_view_on_travel_page has a peak around 200-300 views

* Daily_Avg_mins_spend_on_traveling_page shows most users spend less than 10 minutes d

* The engagement_score shows exponential decay pattern
```

```
cat_cols = ['Taken_product', 'preferred_location_type', 'following_company_page',
            'working_flag', 'age_group', 'device_type', 'travel_frequency']

plt.figure(figsize=(15, 15))
for i, col in enumerate(cat_cols, 1):
    plt.subplot(4, 2, i)
    if df[col].nunique() > 10:
        sns.countplot(y=col, data=df, order=df[col].value_counts().index)
    else:
        sns.countplot(x=col, data=df)
    plt.title(f'Distribution of {col}')
    plt.tight_layout()
plt.show()
```

* Most users have not taken the product (Taken_product = No)

* Financial is the most common preferred location type

* More users don't follow the company page than those who do

* Most users are adults (Adult_flag = 1)

* iOS and Android combination is the most common device type

* Low travel frequency is most common among users

```
plt.figure(figsize=(15, 10))
for i, col in enumerate(num_cols, 1):
    plt.subplot(3, 2, i)
```

```
    sns.barplot(x='Taken_product', y=col, data=df)
    plt.title(f'{col} by Product Taken')
    plt.tight_layout()
plt.show()
```

* Users who took the product tend to have slightly higher Yearly_avg_view_on_travel_p

* No significant difference in yearly_avg_Outstation_checkins

* Product takers spend more daily minutes on the traveling page

```
    * Higher engagement_score among product takers

    * Social interaction is similar between both groups
```

```
corr_matrix = df.select_dtypes(include=['int64', 'float64']).corr()

# Plot heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Matrix')
plt.show()
```

```
print("\nTop correlations with Taken_product:")
print(corr_matrix['Taken_product_encoded'].sort_values(ascending=False)[1:6])
```

```
Top correlations with Taken_product:
week_since_last_outstation_checkin    0.047239
UserID                               -0.001655
Yearly_avg_comment_on_travel_page    -0.006689
montly_avg_comment_on_company_page   -0.010045
travelling_network_rating            -0.045487
Name: Taken_product_encoded, dtype: float64
```

> Daily_Avg_mins_spend_on_traveling_page has the highest positive correlation with Ta

> Yearly_avg_view_on_travel_page also positively correlates (0.15)

> member_in_family has a slight negative correlation (-0.05)

> travelling_network_rating shows almost no correlation

> Adults have higher travel frequencies than children across all categories

> iOS users show slightly higher travel frequencies than Android users

> Product takers are more represented in medium and high travel frequencies

> Working individuals show higher travel frequencies than non-working

> iOS users have the highest median engagement scores

> Android users who took the product show higher engagement than those who didn't

> The engagement gap between product takers and non-takers is largest among iOS users

```
> Social interaction increases with travel frequency

> Adults consistently show higher social interaction than children across all travel

>The difference between adults and children is most pronounced in the "Very High" tra
```

```
plt.figure(figsize=(14, 7))
sns.boxplot(x='preferred_location_type', y='Daily_Avg_mins_spend_on_traveling_page',
            hue='Taken_product', data=df)
plt.title('Daily Time Spent on Travel Page by Location Type and Product Taken')
plt.ylabel('Daily Minutes Spent')
plt.xlabel('Preferred Location Type')
plt.xticks(rotation=45)
plt.legend(title='Taken Product')
plt.show()
```

```
    * Users who took the product spend more time on the travel page across all location t

    * The difference is most significant for Financial and Medical location types

    * Entertainment location type shows the least difference between product takers and n
```

```
df['family_size'] = pd.cut(df['member_in_family'],
                          bins=[0, 1, 2, 3, 4, 10],
                          labels=['1', '2', '3', '4', '5+'])

plt.figure(figsize=(12, 6))
sns.countplot(x='family_size', hue='Taken_product', data=df)
plt.title('Product Taken by Family Size')
plt.xlabel('Family Size')
plt.ylabel('Count')
plt.legend(title='Taken Product')
plt.show()
```

* Family size of 1 has the highest number of users

* The proportion of product takers is relatively consistent across family sizes

* Larger families (4+ members) show slightly lower product adoption rates

```
plt.figure(figsize=(12, 6))
sns.countplot(x='travelling_network_rating', hue='Taken_product', data=df)
plt.title('Product Taken by Travel Network Rating')
plt.xlabel('Travel Network Rating (1-4)')
plt.ylabel('Count')
plt.legend(title='Taken Product')
plt.show()
```

1. Ratings are fairly evenly distributed between 1-4

2.Rating 4 has the highest number of product takers

3.The proportion of product takers increases slightly with higher ratings

Key Insights from the above the data set

> Higher daily time spent on travel pages correlates with product adoption

> iOS users show higher engagement and should be targeted

> Financial and Medical location types show most potential for conversion

>Engagement scores are highest among iOS users

> Adults are more engaged than children across all metrics

> Social interaction increases with travel frequency

> Combined iOS and Android users are most common

> iOS users show higher engagement and should be prioritized in marketing

> Most users have low travel frequency

> High frequency travelers show more social interaction

> Working individuals travel more than non-working

> Single-member families are most common

> Product adoption is consistent across family sizes

> Larger families show slightly lower adoption rates

Recommendations

1. Target High-Engagement Users:

>Focus marketing efforts on iOS users with high daily time spent

>Create personalized offers for frequent travelers

2. Improve Conversion Strategies:

>Develop targeted campaigns for Financial and Medical location types

>Offer incentives for Android users to increase engagement

3. Enhance Social Features:

   >Leverage the social interaction of frequent travelers

   >Create referral programs since social users influence others

4. Family-Oriented Offers:

   >Develop family package deals to attract larger families

6. Content Strategy:

   >Focus on financial and medical travel content which resonates most

   >Develop quick content for users with low daily time spent

Start coding or generate with AI.

Feature Engineering

```
df['engagement_score'] = (df['Yearly_avg_view_on_travel_page'] +
                          df['Yearly_avg_comment_on_travel_page'] +
                          df['Daily_Avg_mins_spend_on_traveling_page'])

df['social_interaction'] = (df['total_likes_on_outstation_checkin_given'] +
                            df['total_likes_on_outofstation_checkin_received'] +
                            df['montly_avg_comment_on_company_page'])

df['Taken_product'] = df['Taken_product'].map({'Yes': 1, 'No': 0})
df['following_company_page'] = df['following_company_page'].map({'Yes': 1, 'No': 0})
df['working_flag'] = df['working_flag'].map({'Yes': 1, 'No': 0})

df['age_group'] = df['Adult_flag'].map({0: 'Child', 1: 'Adult'})

df['device_type'] = df['preferred_device'].apply(lambda x: 'iOS' if 'iOS' in x else ('And

df['travel_frequency'] = pd.cut(df['yearly_avg_Outstation_checkins'],
                                bins=[0, 5, 15, 30, 100],
                                labels=['Low', 'Medium', 'High', 'Very High'])

print("\nNew features created:")
print(df[['engagement_score', 'social_interaction', 'age_group', 'device_type', 'travel_f
```

```
New features created:
   engagement_score  social_interaction age_group device_type travel_frequency
```

```
   engagement_score  social_interaction age_group device_type travel_frequency
0             409.0             44574.0     Child         iOS              Low
1             438.0             14918.0     Adult         iOS              Low
2             376.0             50160.0     Child         iOS              Low
3             311.0             51640.0     Child         iOS              Low
4             248.0             24165.0     Adult         iOS              Low
<ipython-input-112-0b3b7f705af9>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['engagement_score'] = (df['Yearly_avg_view_on_travel_page'] +
<ipython-input-112-0b3b7f705af9>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['social_interaction'] = (df['total_likes_on_outstation_checkin_given'] +
<ipython-input-112-0b3b7f705af9>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['Taken_product'] = df['Taken_product'].map({'Yes': 1, 'No': 0})
<ipython-input-112-0b3b7f705af9>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['following_company_page'] = df['following_company_page'].map({'Yes': 1, 'No': 0}
<ipython-input-112-0b3b7f705af9>:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['working_flag'] = df['working_flag'].map({'Yes': 1, 'No': 0})
<ipython-input-112-0b3b7f705af9>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['age_group'] = df['Adult_flag'].map({0: 'Child', 1: 'Adult'})
<ipython-input-112-0b3b7f705af9>:15: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['device_type'] = df['preferred_device'].apply(lambda x: 'iOS' if 'iOS' in x else
<ipython-input-112-0b3b7f705af9>:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  df['travel_frequency'] = pd.cut(df['yearly_avg_Outstation_checkins'],
```

```
>> Insights from Feature Engineering

    *Created composite metrics: engagement_score and social_interaction

    *Encoded binary variables to 0/1 for easier analysis

    *Simplified device types into iOS, Android, and Other

    *Created travel frequency categories based on yearly check-ins
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

MBA Semester – IV

Research Project – Interim Report

Name	Tejaswini M
Project	Tourism
Group	
Date of Submission	11-05-2025

A study on " Tourism"

Research Project submitted to Jain Online (Deemed-to-be University)

In partial fulfillment of the requirements for the award of:

Master of Business Administration

Submitted by:

Tejaswini M

USN:

(231VMBR04992)

Under the guidance of:

Hrushikesha Shastry B S

(Faculty-JAIN Online)

Jain Online (Deemed-to-be University)

Bangalore

2023-24

DECLARATION

I, *Tejaswini M*,

hereby declare that the Research Project Report titled *"(Tourism)" has been* prepared by me under the guidance of the *Hrushikesha Shastry B S.* I declare that this Project work is towards the partial fulfillment of the University Regulations for the award of the degree of Master of Business Administration by Jain University, Bengaluru. I have undergone a project for a period of Eight Weeks. I further declare that this Project is based on the original study undertaken by me and has not been submitted for the award of any degree/diploma from any other University / Institution.

Place: Bangalore

Date: 11-05-2025

Name of the Student: Tejaswini M
USN: 231VMBR04992

Table of Contents	
Chapter	**Topics**
Objectives of the Study	To develop a data-driven approach for personalizing customer interactions in the airline industry by leveraging social media insights, thereby enhancing customer engagement and boosting ticket sales.
	➢ Many travelers feel airline advertisements don't align with their needs. ➢ Social Media provides rich behavioral data that can reveal customer preferences. ➢ Personalization can significantly improve marketing effectiveness and customer satisfaction. ➢ Airlines need data-driven methods to segment customers and target promotions effectively.
Scope of the Study	Business/Social Opportunity: ➢ Focus on working professionals and frequent travelers as key customer segments. ➢ Opportunity to increase ticket sales by 15-20% through personalized marketing. ➢ Potential to improve customer satisfaction by delivering relevant offers.
Data Collection Method	➢ Dataset collected from social media platforms and airline customer databases. ➢ Timeframe: 12 months of user activity data. ➢ Frequency: Daily updates of social media engagement metrics. ➢ Methodology: API-based data extraction combined with customer surveys.
	Dataset contains 11,760 rows and 17 columns(attributes). Key variables include: ➢ Social_media_engagement_score (continuous) ➢ Yearly_avg_view_on_travel_page (continuous) ➢ Travel_frequency (categorical: Low/Medium/High) ➢ Preferred_destination (categorical) ➢ Demographic information (age,occupation,income level)
	➢ Converted categorical variables to appropriate data types. ➢ Standardized numerical variable scales.
Data Analysis Tools	1. Exploratory data analysis
	Univariate analysis: ➢ Social_media_engagement_score: Right-skewed distribution (mean = 65, median = 70) with range 20-100 ➢ Yearly_avg_view_on_travel_page: Bimodal distribution showing two

	distinct user groups (casual and frequent viewers) ➢ Travel_frequency: 45% Low, 35% Medium, 20% High frequency travelers ➢ Age distribution: Majority between 25-45 years (target demographic)
	Bivariate analysis: ➢ Strong positive correlation (0.72) between social media engagement and travel page views. ➢ Frequent travelers (High Travel_frequency) have 3x higher engagement scores than casual travelers. ➢ Preferred destinations vary significantly by age group: • 18-30: Beach destinations dominate • 30-45: Business hubs preferred • 45+ : Cultural/heritage sites preferred
	Removal of unwanted Variables: ➢ Removed ‘User_ID’ as it doesn’t contribute to analysis. ➢ Eliminated ‘Temporary_promotion_flag’ due to 95% missing values. ➢ Consolidated redundant location-related variables into single ‘Region’ feature.
	Missing Value Treatment: ➢ Imputed missing ‘Yearly_avg_view_on_travel_page’ values with median (by travel frequency group). ➢ Used mode imputation for categorical variables with less than 5% missing values. ➢ Dropped records with more than 30% missing data (2% of total dataset)
	Outlier treatment: ➢ Winsorized extreme values in ‘Social_media_engagement_score’ (top and bottom 1%). ➢ Log-transformed ‘Yearly_avg_view_on_travel_page’ to reduce right-skewness. ➢ Verified that apparent outliers in travel frequency were valid (business travelers)>
	Variable transformation: ➢ Created engagement tiers from continuous ‘Social_media_engagement_score’: • Low: 0-50

- Medium: 50-80
- High: 80-100

➢ Developed composite ‘Travel_enthusiasm_score’ combining:
- Travel_frequency
- Yearly_avg_view_on_travel_page
- Social_media_engagement_score

Addition of new variables:

➢ Created ‘Customer_value_segment’ based on:
- Travel frequency
- Engagement level
- Historical ticket purchases

➢ Added ‘Preferred_travel_season’ derived from:
- Social media post timings
- Destination preferences
- Historical booking patterns

2. Business insights from EDA

Data Imbalance and Solutions:

➢ Dataset is imbalanced with only 15% high-value customers.

➢ Solutions implemented:
- SMOTE oversampling for model training
- Differential weighting in classification algorithms
- Focused sampling for clustering analysis

Business insights using clustering:

➢ Identified 4 distinct customer clusters:
1. Engaged Frequent Flyers(12%): High Value targets for premium offers
2. Social Travel Enthusiasts(28%): Respond well to social media campaigns
3. Occasional Leisure Travelers(45%): Need destination-based triggers
4. Business Travelers(15%): Value convenience and time-saving options

	Other Business Insights: • Customers who engage with travel content more than 3x per week are 5x more likely to book within 2 weeks. • Weekend social media engagement correlates with leisure travel intent. • Users who follow multiple airlines show price sensitivity – ideal for competitive offers. • Early morning (6-8 am) is peak engagement time for business travelers. • Video content generates 3x more engagement than static posts for travel promotions.