data mining final project

February 6, 2025

[1]:
```
# Adding necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
from imblearn.over_sampling import SMOTE
import warnings

# Suppress warnings
warnings.filterwarnings("ignore")
```

[2]:
```
# Loading the dataset
hotel_df = pd.read_csv('filtrado.csv')
```

[3]:
```
# Information about the dataset
print("Dataset Info:")
hotel_df.info()
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117263 entries, 0 to 117262
Data columns (total 33 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unnamed: 0                      117263 non-null  int64
 1   hotel                           117263 non-null  object
 2   is_canceled                     117263 non-null  int64
 3   lead_time                       117263 non-null  int64
```

```
 4   arrival_date_year               117263 non-null  int64
 5   arrival_date_month              117263 non-null  object
 6   arrival_date_week_number        117263 non-null  int64
 7   arrival_date_day_of_month       117263 non-null  int64
 8   stays_in_weekend_nights         117263 non-null  int64
 9   stays_in_week_nights            117263 non-null  int64
 10  adults                          117263 non-null  int64
 11  children                        117263 non-null  float64
 12  babies                          117263 non-null  int64
 13  meal                            117263 non-null  object
 14  country                         117263 non-null  object
 15  market_segment                  117263 non-null  object
 16  distribution_channel            117263 non-null  object
 17  is_repeated_guest               117263 non-null  int64
 18  previous_cancellations          117263 non-null  int64
 19  previous_bookings_not_canceled  117263 non-null  int64
 20  reserved_room_type              117263 non-null  object
 21  assigned_room_type              117263 non-null  object
 22  booking_changes                 117263 non-null  int64
 23  deposit_type                    117263 non-null  object
 24  agent                           117263 non-null  float64
 25  company                         117263 non-null  float64
 26  days_in_waiting_list            117263 non-null  int64
 27  customer_type                   117263 non-null  object
 28  adr                             117263 non-null  float64
 29  required_car_parking_spaces     117263 non-null  int64
 30  total_of_special_requests       117263 non-null  int64
 31  reservation_status              117263 non-null  object
 32  reservation_status_date         117263 non-null  object
dtypes: float64(4), int64(17), object(12)
memory usage: 29.5+ MB
```

[4]:
```
# Head diplays first 5 rows
print("\nFirst 5 Rows:")
print(hotel_df.head())
```

```
First 5 Rows:
   Unnamed: 0         hotel  is_canceled  lead_time  arrival_date_year  \
0           2  Resort Hotel            0          7               2015
1           3  Resort Hotel            0         13               2015
2           4  Resort Hotel            0         14               2015
3           5  Resort Hotel            0         14               2015
4           6  Resort Hotel            0          0               2015

  arrival_date_month  arrival_date_week_number  arrival_date_day_of_month  \
0               July                        27                          1
1               July                        27                          1
```

```
2               July                        27                          1
3               July                        27                          1
4               July                        27                          1

   stays_in_weekend_nights  stays_in_week_nights  …  deposit_type  agent  \
0                        0                     1  …    No Deposit    0.0
1                        0                     1  …    No Deposit  304.0
2                        0                     2  …    No Deposit  240.0
3                        0                     2  …    No Deposit  240.0
4                        0                     2  …    No Deposit    0.0

   company days_in_waiting_list customer_type    adr  \
0      0.0                     0     Transient   75.0
1      0.0                     0     Transient   75.0
2      0.0                     0     Transient   98.0
3      0.0                     0     Transient   98.0
4      0.0                     0     Transient  107.0

  required_car_parking_spaces  total_of_special_requests  reservation_status  \
0                           0                          0           Check-Out
1                           0                          0           Check-Out
2                           0                          1           Check-Out
3                           0                          1           Check-Out
4                           0                          0           Check-Out

   reservation_status_date
0               2015-07-02
1               2015-07-02
2               2015-07-03
3               2015-07-03
4               2015-07-03

[5 rows x 33 columns]
```

[5]:
```
# Dropping the 'Unnamed: 0' column as it seems to be an index column with no␣
 ↪approproiate meaning
hotel_df.drop(columns=['Unnamed: 0'], inplace=True)
```

[6]:
```
#  Check for missing values (if any)
hotel_df.isna().sum()
```

[6]:
```
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                          0
babies                            0
meal                              0
country                           0
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                             0
company                           0
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
dtype: int64
```

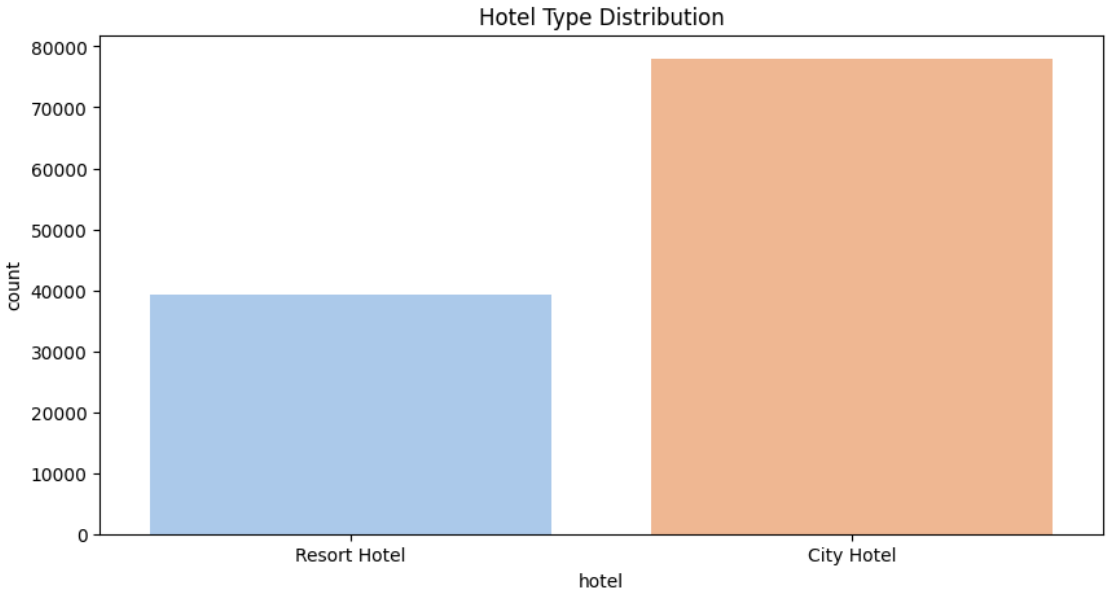
[7]:
```
# Statistical summary
print("\nSummary Statistics:")
print(hotel_df.describe())
```

```
Summary Statistics:
         is_canceled      lead_time  arrival_date_year  \
count  117263.000000  117263.000000      117263.000000
mean        0.373801     104.479887        2016.160664
std         0.483814     106.946412           0.706854
min         0.000000       0.000000        2015.000000
25%         0.000000      18.000000        2016.000000
50%         0.000000      70.000000        2016.000000
75%         1.000000     161.000000        2017.000000
max         1.000000     709.000000        2017.000000

       arrival_date_week_number  arrival_date_day_of_month  \
count             117263.000000              117263.000000
mean                  27.142679                  15.807407
std                   13.576785                   8.784655
```

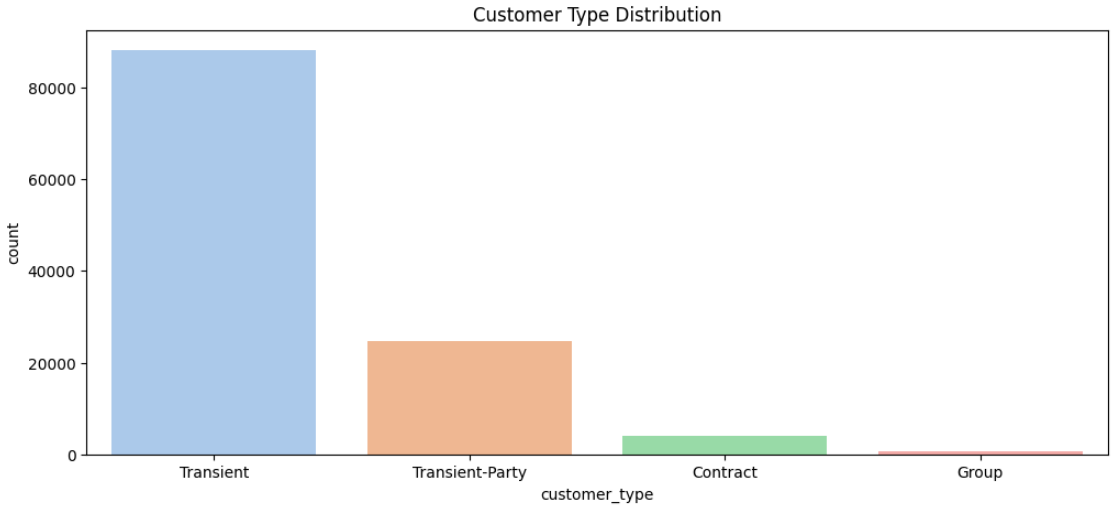
```
min                   1.000000                  1.000000
25%                  16.000000                  8.000000
50%                  27.000000                 16.000000
75%                  38.000000                 23.000000
max                  53.000000                 31.000000

       stays_in_weekend_nights  stays_in_week_nights         adults  \
count            117263.000000         117263.000000  117263.000000
mean                  0.929117              2.505095       1.855334
std                   0.999236              1.909409       0.494222
min                   0.000000              0.000000       0.000000
25%                   0.000000              1.000000       2.000000
50%                   1.000000              2.000000       2.000000
75%                   2.000000              3.000000       2.000000
max                  19.000000             50.000000      20.000000

            children         babies  is_repeated_guest  \
count  117263.000000  117263.000000      117263.000000
mean        0.104313       0.007914           0.031382
std         0.399511       0.097410           0.174350
min         0.000000       0.000000           0.000000
25%         0.000000       0.000000           0.000000
50%         0.000000       0.000000           0.000000
75%         0.000000       0.000000           0.000000
max        10.000000      10.000000           1.000000

       previous_cancellations  previous_bookings_not_canceled  \
count           117263.000000                   117263.000000
mean                0.086199                        0.130723
std                 0.840916                        1.443521
min                 0.000000                        0.000000
25%                 0.000000                        0.000000
50%                 0.000000                        0.000000
75%                 0.000000                        0.000000
max                26.000000                       72.000000

       booking_changes          agent        company  days_in_waiting_list  \
count    117263.000000  117263.000000  117263.000000         117263.000000
mean          0.219967      74.890724      10.684871              2.325474
std           0.647851     107.162561      53.712670             17.661022
min           0.000000       0.000000       0.000000              0.000000
25%           0.000000       7.000000       0.000000              0.000000
50%           0.000000       9.000000       0.000000              0.000000
75%           0.000000     152.000000       0.000000              0.000000
max          21.000000     535.000000     543.000000            391.000000

                adr  required_car_parking_spaces  total_of_special_requests
count  117263.000000                117263.000000              117263.000000
```

mean	102.100844	0.062279	0.571007
std	50.580517	0.244922	0.793045
min	-6.380000	0.000000	0.000000
25%	70.000000	0.000000	0.000000
50%	95.000000	0.000000	0.000000
75%	126.000000	0.000000	1.000000
max	5400.000000	8.000000	5.000000

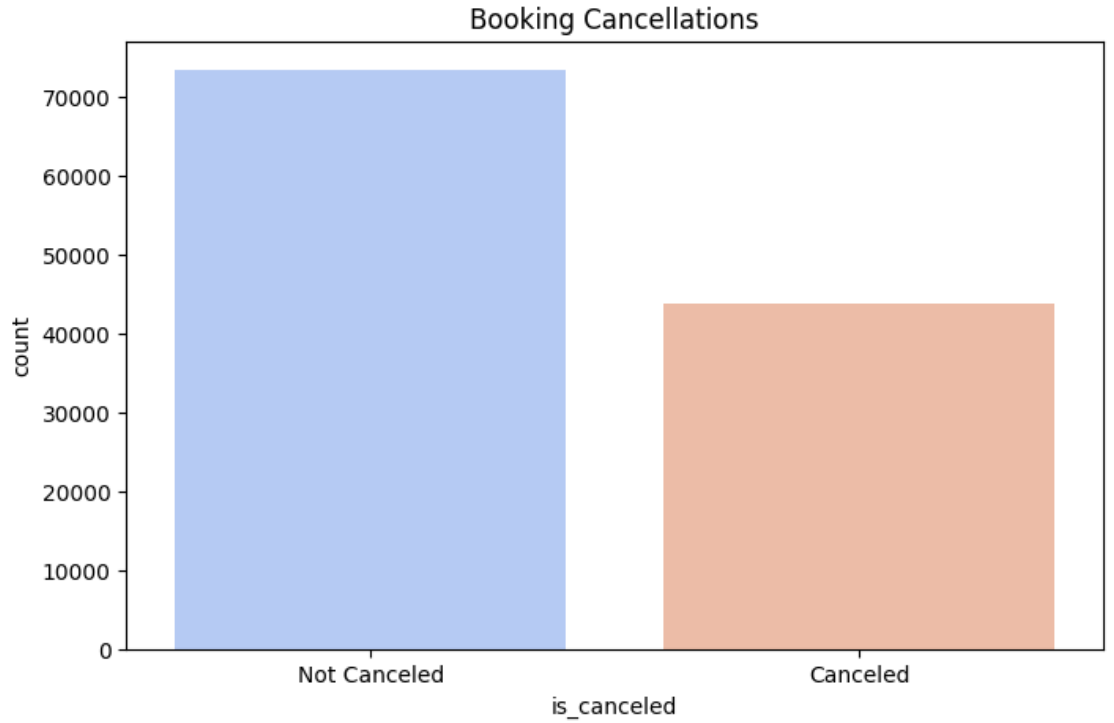
[8]:
```
# Visualizing categorical variable distribution
plt.figure(figsize=(10, 5))
sns.countplot(data=hotel_df, x='hotel', palette='pastel')
plt.title('Hotel Type Distribution')
plt.show()
```



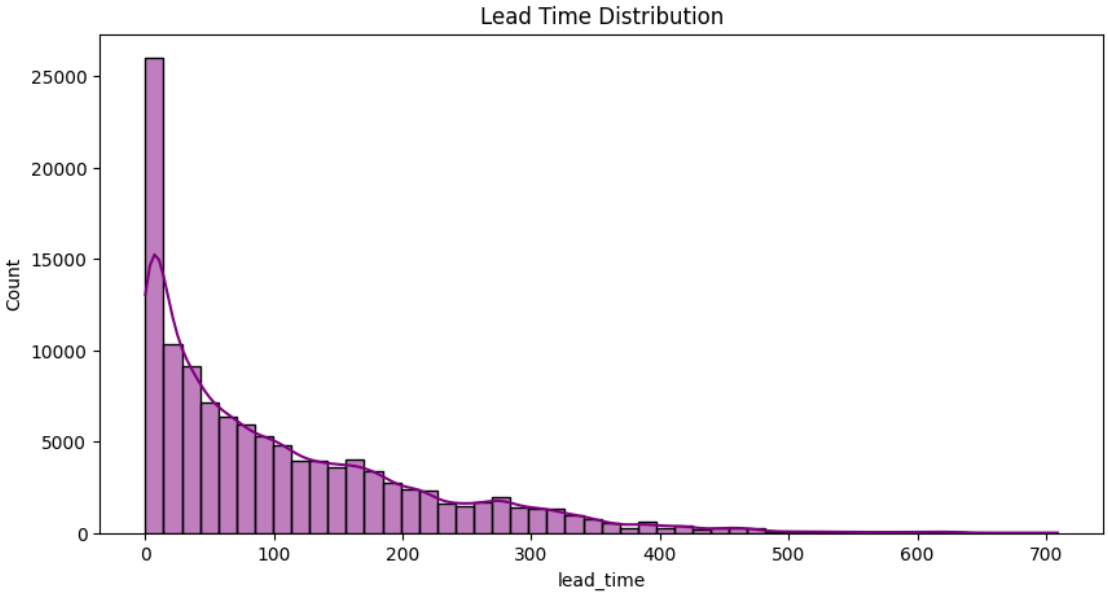
[9]:
```
plt.figure(figsize=(12, 5))
sns.countplot(data=hotel_df, x='customer_type', palette='pastel',␣
↪order=hotel_df['customer_type'].value_counts().index)
plt.title('Customer Type Distribution')
plt.show()
```



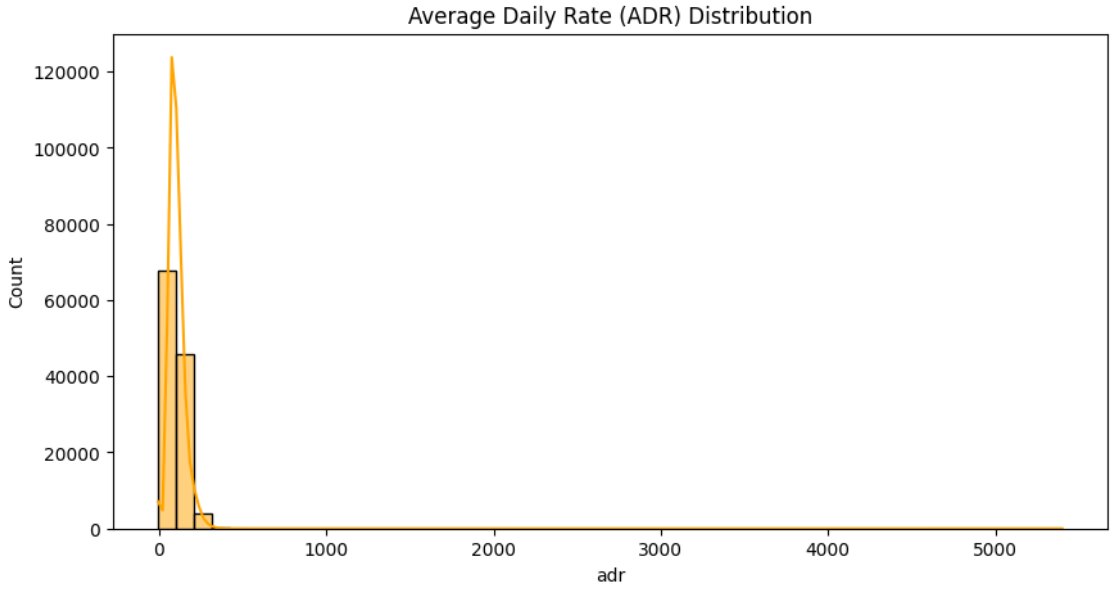
[10]:
```
# Visualizing cancellation rates
plt.figure(figsize=(8, 5))
sns.countplot(data=hotel_df, x='is_canceled', palette='coolwarm')
plt.title('Booking Cancellations')
plt.xticks(ticks=[0, 1], labels=['Not Canceled', 'Canceled'])
plt.show()
```



[11]:
```
# Distribution of lead time
plt.figure(figsize=(10, 5))
sns.histplot(hotel_df['lead_time'], bins=50, kde=True, color='purple')
plt.title('Lead Time Distribution')
plt.show()
```

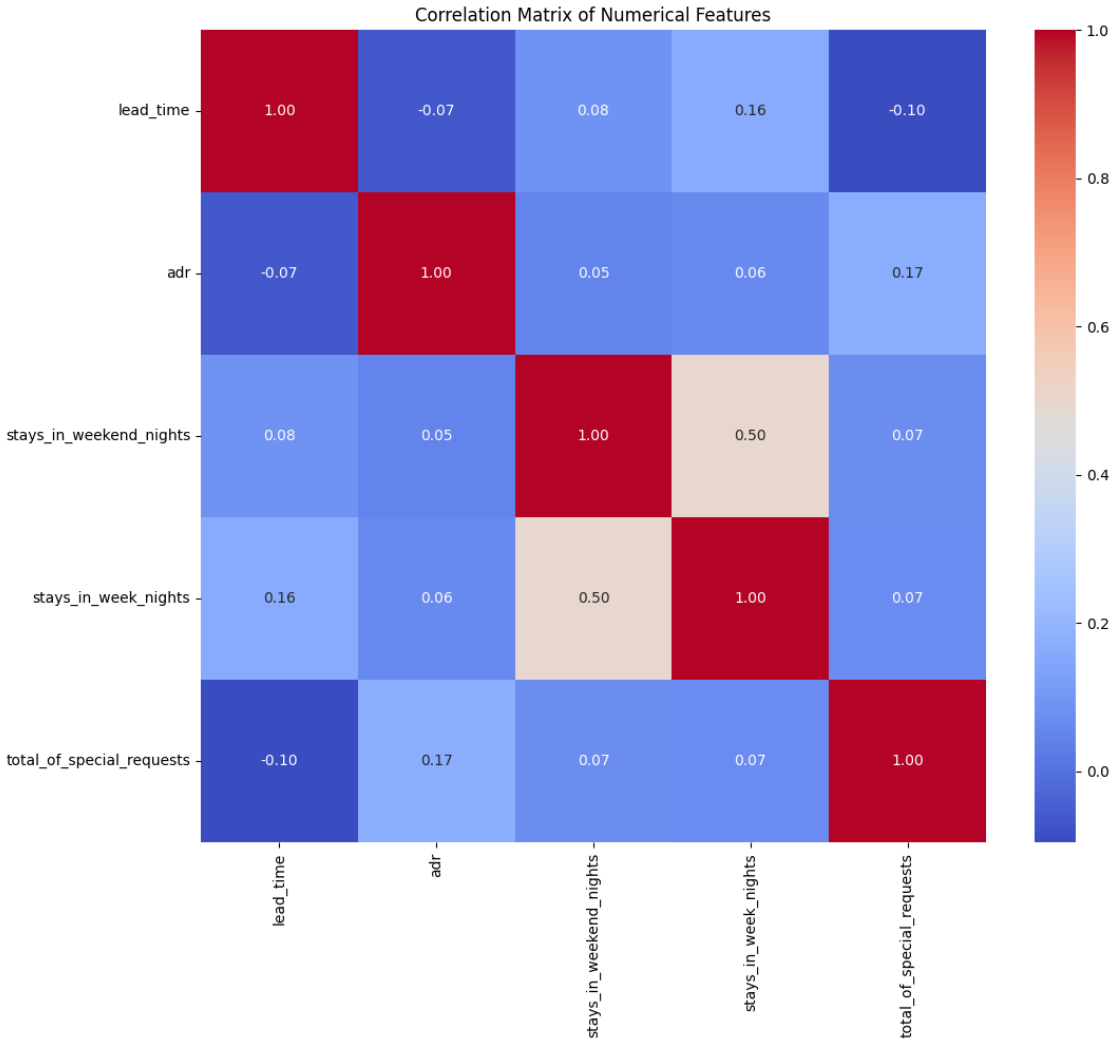


[12]:
```
# Average daily rate distribution
plt.figure(figsize=(10, 5))
sns.histplot(hotel_df['adr'], bins=50, kde=True, color='orange')
plt.title('Average Daily Rate (ADR) Distribution')
plt.show()
```

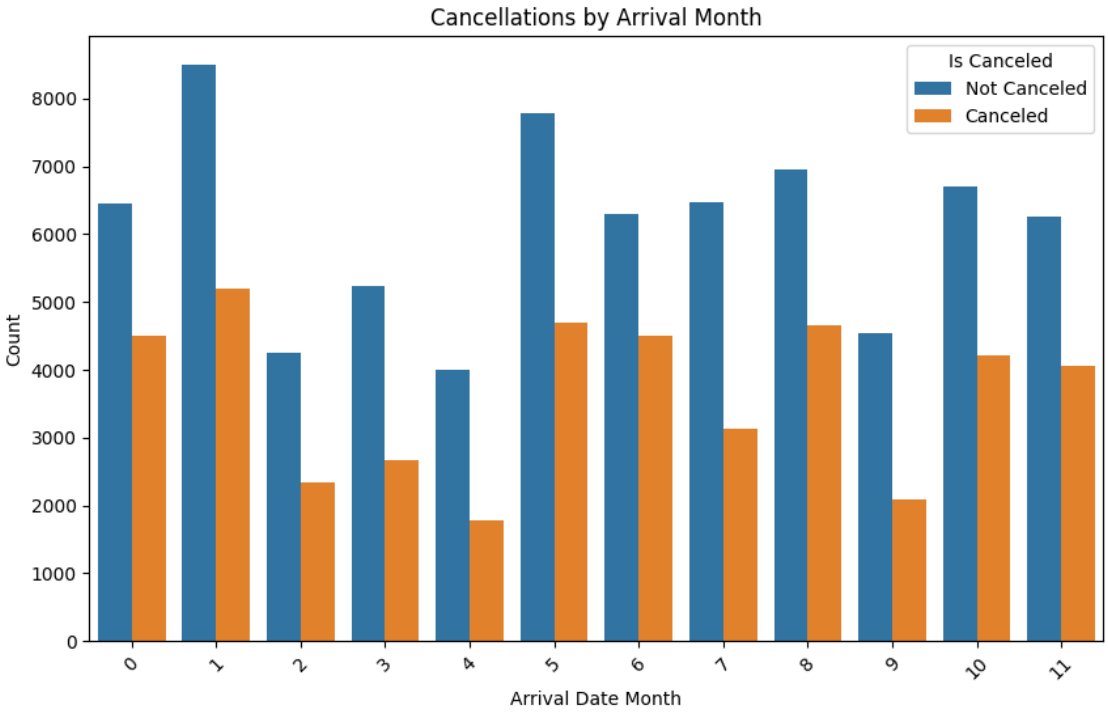


[13]:
```
# Encoding categorical variables
categorical_cols = ['hotel', 'arrival_date_month', 'meal', 'country',␣
 ↪'market_segment',
                    'distribution_channel', 'reserved_room_type',␣
 ↪'assigned_room_type',
                    'deposit_type', 'customer_type', 'reservation_status',␣
 ↪'reservation_status_date']

label_encoder = LabelEncoder()
for col in categorical_cols:
    hotel_df[col] = label_encoder.fit_transform(hotel_df[col])
```

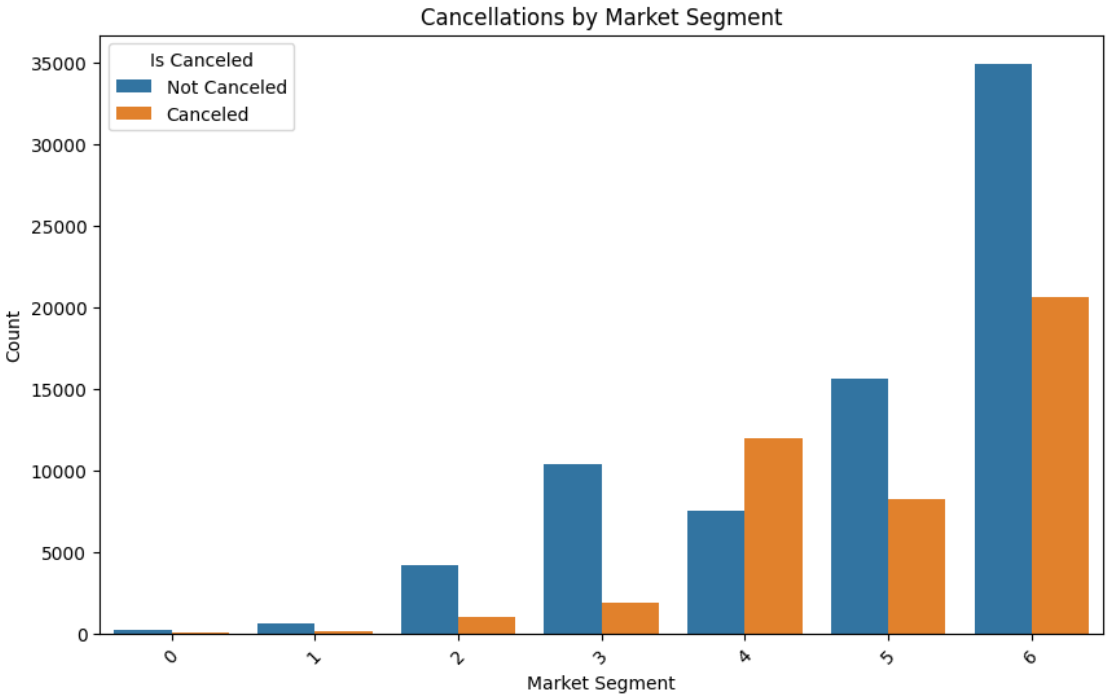
[14]:
```
# Correlation matrix of numerical features
numerical_cols = ['lead_time', 'adr', 'stays_in_weekend_nights',␣
 ↪'stays_in_week_nights', 'total_of_special_requests']
plt.figure(figsize=(12, 10))
correlation_matrix = hotel_df[numerical_cols[:5]].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',fmt='.2f')
plt.title('Correlation Matrix of Numerical Features')
plt.show()
```



[15]:
```
# Plotting the bar chart of 'arrival_date_month' against the count of␣
 ↪cancellations
plt.figure(figsize=(10, 6))
sns.countplot(x='arrival_date_month', hue='is_canceled', data=hotel_df)
plt.title('Cancellations by Arrival Month')
plt.xlabel('Arrival Date Month')
plt.ylabel('Count')
plt.legend(title='Is Canceled', labels=['Not Canceled', 'Canceled'])
plt.xticks(rotation=45)
plt.show()
```

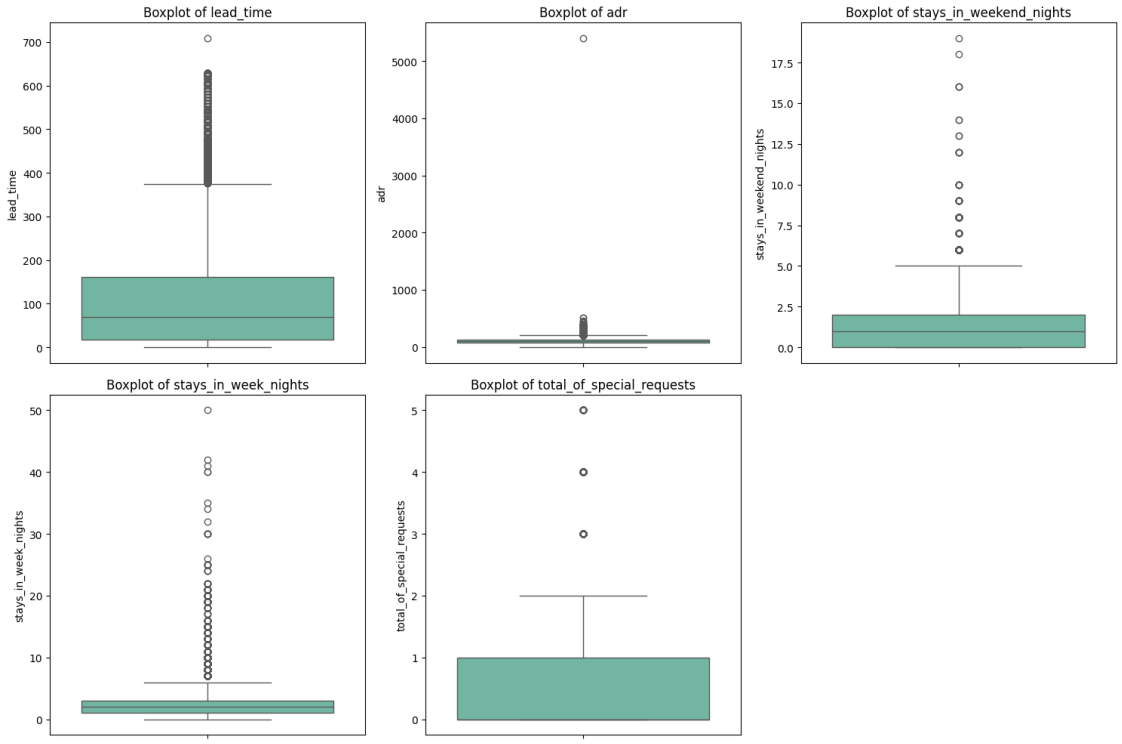


[16]:
```
# Plotting the bar chart of 'market_segment' against the count of cancellations
plt.figure(figsize=(10, 6))
sns.countplot(x='market_segment', hue='is_canceled', data=hotel_df)
plt.title('Cancellations by Market Segment')
plt.xlabel('Market Segment')
plt.ylabel('Count')
plt.legend(title='Is Canceled', labels=['Not Canceled', 'Canceled'])
plt.xticks(rotation=45)
plt.show()
```



1 Outlier detection

[17]:
```
plt.figure(figsize=(15, 10))
for i, col in enumerate(numerical_cols, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(y=hotel_df[col], palette='Set2')
    plt.title(f'Boxplot of {col}')
plt.tight_layout()
plt.show()
```



[18]:
```
# Removing outliers using the IQR method
for col in numerical_cols:
    Q1 = hotel_df[col].quantile(0.25)
    Q3 = hotel_df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    hotel_df = hotel_df[(hotel_df[col] >= lower_bound) & (hotel_df[col] <=␣
 ↪upper_bound)]

print("\nOutliers removed, updated dataset shape:", hotel_df.shape)
```

```
Outliers removed, updated dataset shape: (104986, 32)
```

1.0.1 The above visulizations provides an understanding of how variables are related to each other.

1.0.2 The EDA covers like dataset loading, checking misssing values, statistical summary distribution,outlier detection.

1.0.3 In the next phase I will start data preprocessing and model implementation

[19]:
```
# Splitting Data into Train and Test Sets
X = hotel_df.drop(columns=["is_canceled","reservation_status"])
y = hotel_df["is_canceled"]  # Target variable

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=42)
```

[20]:
```
# Scaling Numerical Features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[21]:
```
hotel_df["is_canceled"].value_counts()
```

[21]:
```
is_canceled
0    66067
1    38919
Name: count, dtype: int64
```

1.1 As we can see from above data is bias towards not cancelled. So I implemnted SMOTE technique to balance the data by creating synthetic samples.

[22]:
```
# SMOTE for Class Balancing
smote = SMOTE(sampling_strategy='auto', random_state=42)
X_train_balanced, y_train_balanced = smote.fit_resample(X_train, y_train)

# new class distribution after SMOTE
print("Class Distribution After SMOTE:")
print(pd.Series(y_train_balanced).value_counts())
```

```
Class Distribution After SMOTE:
is_canceled
1    46314
0    46314
Name: count, dtype: int64
```

[23]:
```
# Applying Multiple ML Models
models = {
    "Random Forest": RandomForestClassifier(n_estimators=100, max_depth=10,␣
 ↪random_state=42),
    "Logistic Regression": LogisticRegression(max_iter=500),
```

```
    "SVM": SVC(kernel="linear", C=0.5),
    "Decision Tree": DecisionTreeClassifier(max_depth=5, random_state=42),
    "KNN": KNeighborsClassifier(n_neighbors=5)
}
```

[24]:
```
# Dictionary to store the model results
balanced_model_results = {}

for model_name, model in models.items():
    model.fit(X_train_balanced, y_train_balanced)  # Training on SMOTE-balanced
 ↪data
    y_pred_balanced = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred_balanced)

    balanced_model_results[model_name] = {
        "Accuracy": accuracy,
        "Classification Report": classification_report(y_test, y_pred_balanced),
        "Confusion Matrix": confusion_matrix(y_test, y_pred_balanced)
    }
```

[25]:
```
# Converting results into DataFrame
balanced_accuracy_df = pd.DataFrame(
    {model: [result["Accuracy"]] for model, result in balanced_model_results.
 ↪items()}
).T.reset_index()
balanced_accuracy_df.columns = ["Model", "Accuracy"]
```

[26]:
```
# Displaying results
print("Model Accuracies:\n", balanced_accuracy_df)
```

```
Model Accuracies:
                 Model  Accuracy
0        Random Forest  0.867189
1  Logistic Regression  0.957423
2                  SVM  0.958788
3        Decision Tree  0.782925
4                  KNN  0.817405
```

[27]:
```
# Classification report and confusion matrix for each model after applying SMOTE
for model_name, result in balanced_model_results.items():
    print(f"\n===== {model_name} (After SMOTE) =====")
    print(f"Accuracy: {result['Accuracy']:.2f}\n")
    print("Classification Report:\n", result["Classification Report"])
    print("Confusion Matrix:\n", result["Confusion Matrix"])
```

```
===== Random Forest (After SMOTE) =====
Accuracy: 0.87
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.90      0.89     19753
           1       0.82      0.82      0.82     11743

    accuracy                           0.87     31496
   macro avg       0.86      0.86      0.86     31496
weighted avg       0.87      0.87      0.87     31496

Confusion Matrix:
 [[17713  2040]
 [ 2143  9600]]

===== Logistic Regression (After SMOTE) =====
Accuracy: 0.96

Classification Report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97     19753
           1       0.99      0.89      0.94     11743

    accuracy                           0.96     31496
   macro avg       0.97      0.94      0.95     31496
weighted avg       0.96      0.96      0.96     31496

Confusion Matrix:
 [[19668    85]
 [ 1256 10487]]

===== SVM (After SMOTE) =====
Accuracy: 0.96

Classification Report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97     19753
           1       0.99      0.89      0.94     11743

    accuracy                           0.96     31496
   macro avg       0.97      0.95      0.95     31496
weighted avg       0.96      0.96      0.96     31496

Confusion Matrix:
 [[19700    53]
 [ 1245 10498]]
```

```
===== Decision Tree (After SMOTE) =====
Accuracy: 0.78

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.83      0.83     19753
           1       0.71      0.71      0.71     11743

    accuracy                           0.78     31496
   macro avg       0.77      0.77      0.77     31496
weighted avg       0.78      0.78      0.78     31496

Confusion Matrix:
 [[16325  3428]
 [ 3409  8334]]

===== KNN (After SMOTE) =====
Accuracy: 0.82

Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.81      0.85     19753
           1       0.72      0.83      0.77     11743

    accuracy                           0.82     31496
   macro avg       0.81      0.82      0.81     31496
weighted avg       0.83      0.82      0.82     31496

Confusion Matrix:
 [[16005  3748]
 [ 2003  9740]]
```

1.1.1 The model is created with several machine learning algorithms and out of those Logisitic regression and SVM performing very well.