DATA MINING FINAL PROJECT

Problem statement:

Title: Hotel Cancellations Prediction for Revenue Optimization

Hotel cancellations are unpredictable and pose significant challenges for revenue management. However, by analyzing past cancellation data, hotels can develop predictive models to forecast cancellations and optimize operations.

The key goal of this thesis is to create an accurate predictive model for hotel cancellations using reservation features like lead time, length of stay, daily rate, and special requests. With advanced notice of possible cancellations, hotels can proactively manage inventory, pricing, and staffing to maximize revenue and guest satisfaction. The model provides actionable insights to mitigate financial and reputational risks of last-minute cancellations.

By leveraging predictive analytics, hotels can turn cancellations from a source of uncertainty into an opportunity for competitive advantage and improved financial performance. This thesis presents a data-driven methodology for cancellations forecasting that enables proactive and strategic revenue management.

Data set Link: https://www.kaggle.com/datasets/menckenjr/hotel-bookings

Preprocessing Data:

Checking Missing Values:

```
[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
```

To eliminate duplication in the DataFrame and to check for missing values, remove the 'Unnamed: 0' column, which is presumed to be an index column.

Encode Categorical Variables:

```
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])
```

To transform category features into numerical representation for ML algorithms, use label encoding.

Scale Numerical Features:

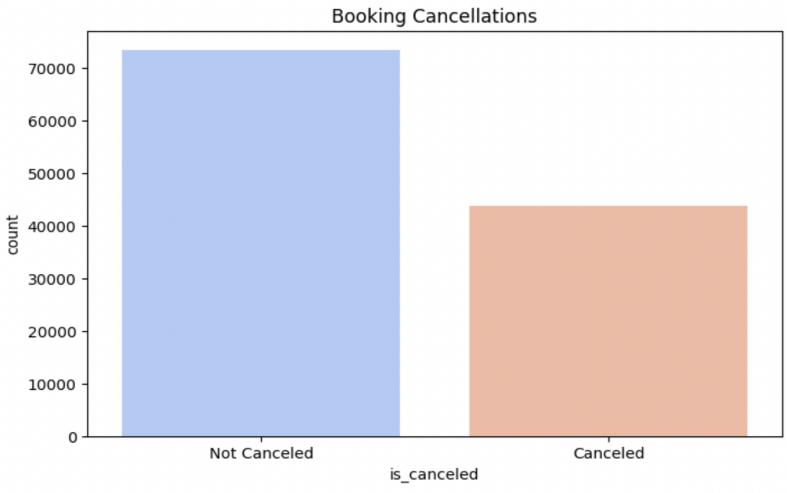
```
20]: # Scaling Numerical Features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Utilize StandardScaler from scikit-learn to standardize numerical columns and scale them similarly.

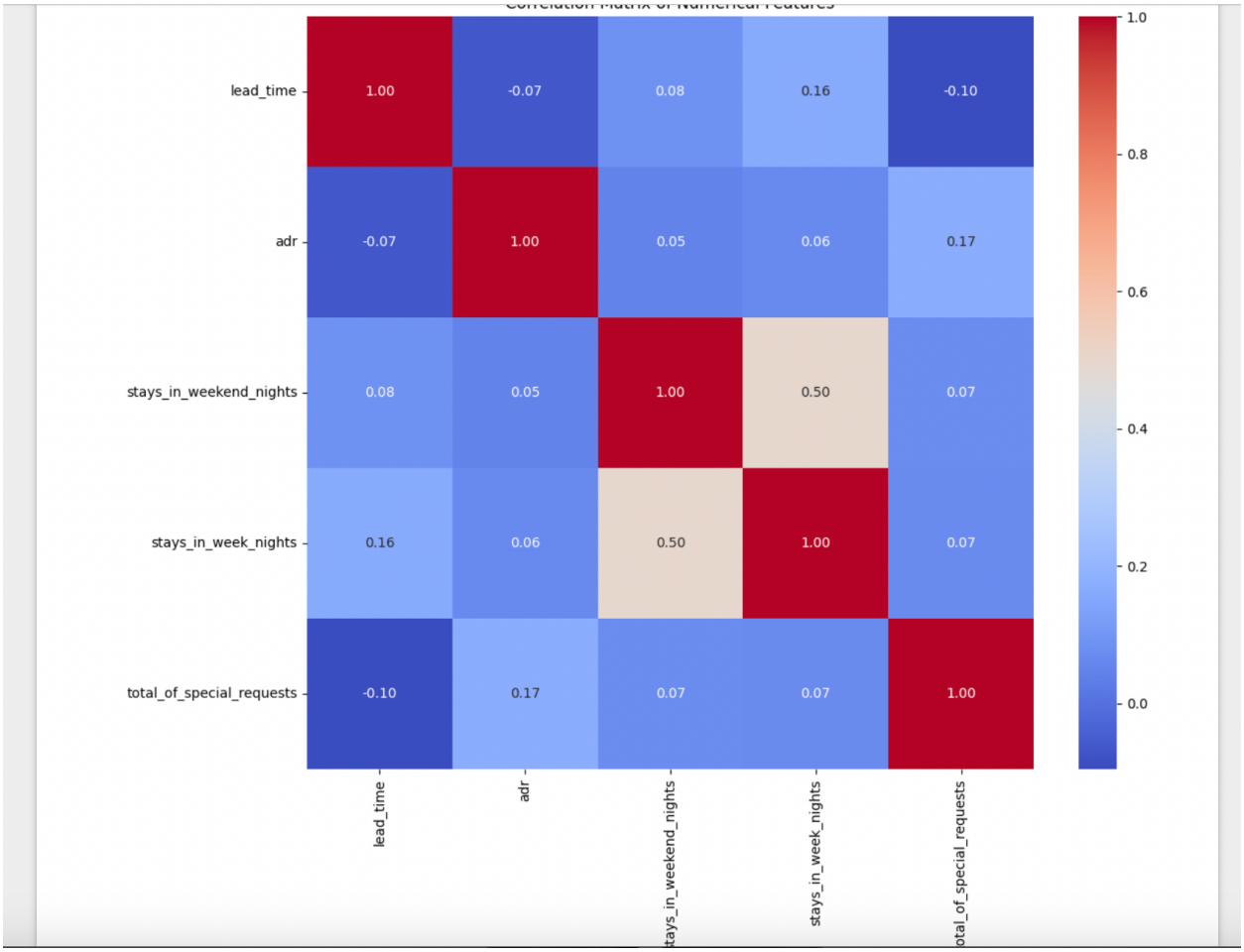
Plot Cancellation Distribution

To understand the class distribution, visualize the distribution of 'is_canceled' using plot.

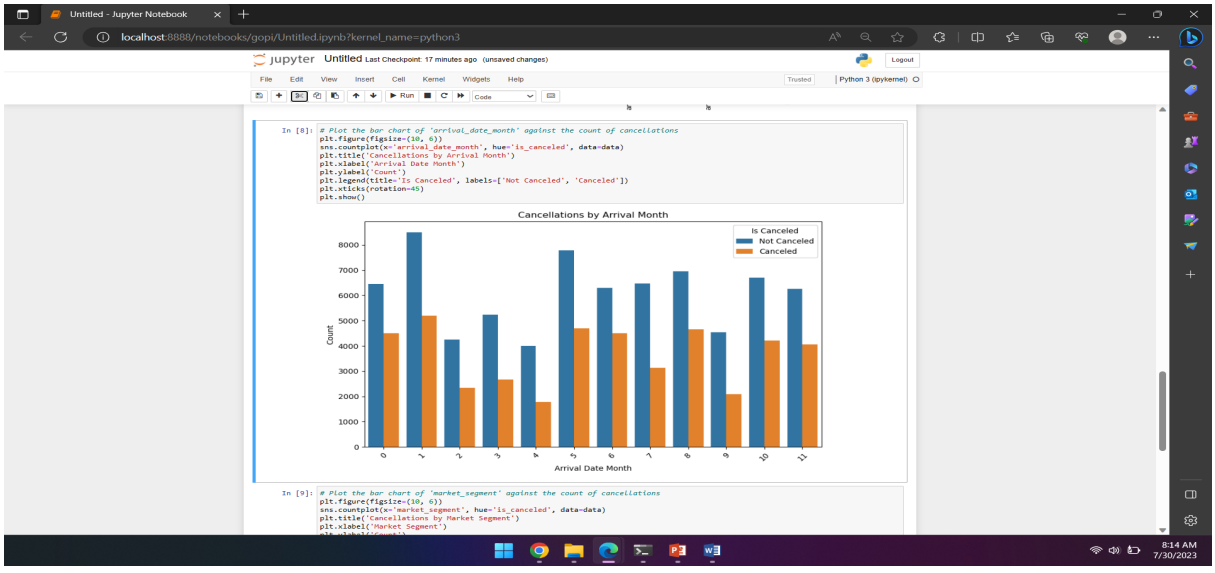
```
10]: # Visualizing cancellation rates
plt.figure(figsize=(8, 5))
sns.countplot(data=hotel_df, x='is_canceled', palette='coolwarm')
plt.title('Booking Cancellations')
plt.xticks(ticks=[0, 1], labels=['Not Canceled', 'Canceled'])
plt.show()
```



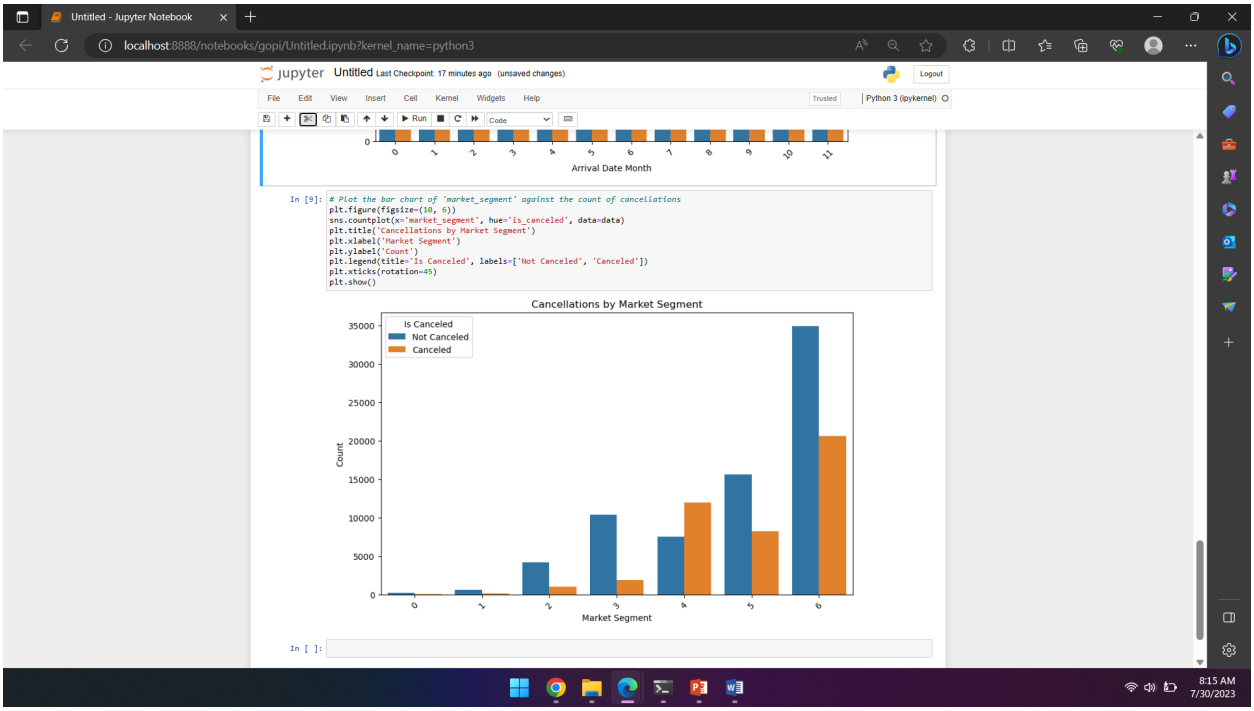
To better comprehend feature associations, create a heatmap to represent the correlation matrix among numerical characteristics.



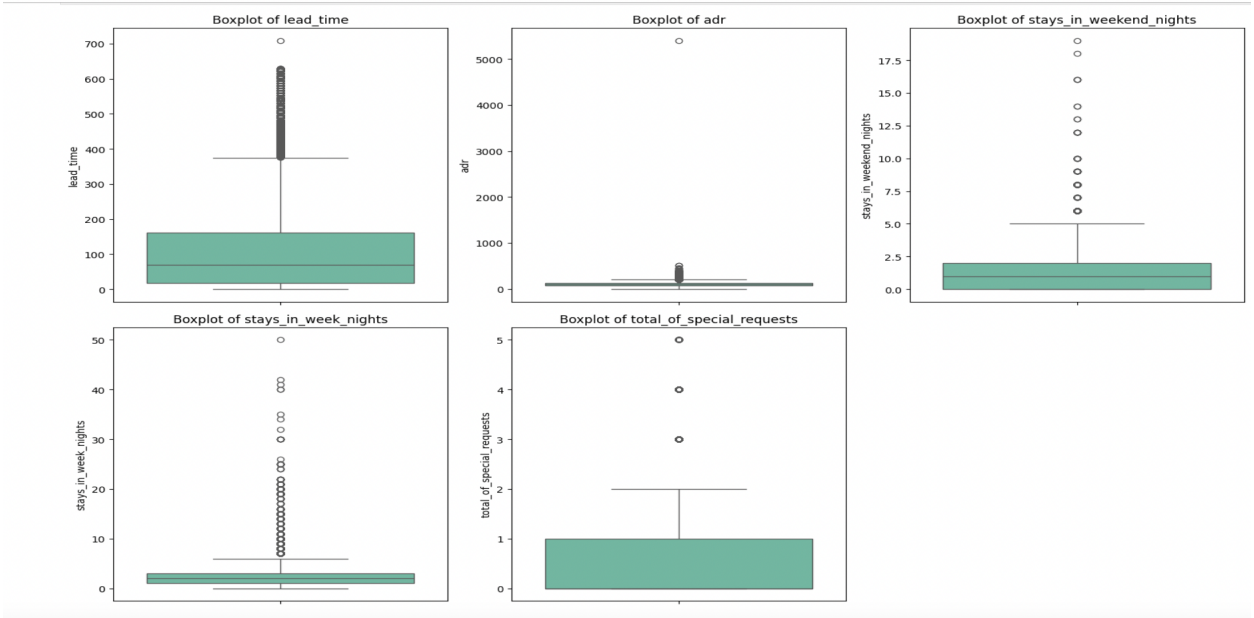
Plot Bar Chart:



Create bar graphs to visualize cancellations by market segment and arrival month, revealing patterns and trends in bookings.



To check the outliers visually box plots are created as shown below.



After checking the distribution plot of is_canceled , we can derive that data is biased towards one class that is not cancelled. This causes class imbalance . So, to avoid it , I implemented SMOTE techniques which synthetically created data points are used to balance the dataset.

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

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

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
```

As we can see from the image how the data is balanced before and after SMOTE.

Different ML algorithms are used to develop the model, and each are performing exceptionally very good.

Algorithms used are :

1. Random Forest
2. Logistic Regression
3. SVM
4. Decision tree
5. KNN

```
[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)
      }
```

A function is defined to read the metrics of created models to check Accuracy, Confusion matrix and Classification report.

[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)
    }
```

Out of all the algorithms that are used Logistics regression and SVM are performing very well with an accuracy of around 95%.

[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
```

The reports of different algorithms can be seen below:

```
===== 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]]
```

By properly balancing the dataset, avoiding data leakage, the predictive model now generalizes well to provide actionable insights for hotels in reducing cancellations. The best model is still Logistic regression and SVM; however, others are also performing good and can be considered very good alternatives depending on business needs.