Technical Appendix

1. Data Description

1.1 Data Source and Collection

The dataset used in this analysis was obtained from **Kaggle's "Expedia Hotel Booking Behavior"** dataset, a publicly available secondary data source used for academic and analytical research purposes. The dataset contains over 9.9 million user search records from an online travel platform, and includes information on search behavior, hotel characteristics, promotions, and booking outcomes. Because of its large size, a stratified random sample of 1 million rows was selected to ensure computational efficiency while maintaining the overall booking distribution. No personal identifiable information was included, and no API, web-scraping, or special permissions were required. The data was downloaded directly from Kaggle and used under its standard open data usage permissions.

1.2 Variables List

Variable Name	Description	Type	Transformation/Notes
booking_bool	Booking outcome (1/0)	Binary	Target variable
price_usd	Nightly hotel price	Numeric	StandardScaled
srch_length_of_stay	Length of stay in nights	Numeric	Scaled
srch_adults_count	Number of adults in the search	Numeric	Used for segmentation
srch_children_count	Number of children	Numeric	Indicates family/group
srch_room_count	Rooms requested	Numeric	Scaled
orig_destination_distance	Distance between user and destination	Numeric	PowerTransformer applied
prop_starrating	Star rating of property	Numeric	Used as quality indicator
prop_review_score	User review score	Numeric	Normalized
promotion_flag	Whether a promotion was shown	Binary	Used for promo-sensitivity analysis
srch_saturday_night_bool	Saturday-night included	Binary	Used to infer trip length

From the original 54 variables, a subset of behavioral, hotel quality, and promotional features was selected for clustering and regression analysis. After data cleaning, transformation, and scaling, the final dataset was used to generate four distinct customer segments and analyze booking behaviors within each segment.

2. Data Processing

2.1 Data Cleaning Procedures

- Data Cleaning:
  - Check each column for missing values and remove them where necessary.
  - Select only numeric columns and use the extreme outlier capping method to detect outliers in each column. Remove extreme outliers to maintain data integrity and modeling accuracy.

- Summary of integrity checks and filters:
  - Check that each column was within valid ranges and has proper data types (i.e., float for price and int for prop_starrating)
  - Filter out unrelated columns and retain only the relevant variables required for analysis (11 relevant variables were selected from the original 54 features)
  - Validate that null values were appropriately handled prior to transformation
  - Conduct a final consistency review to confirm all retained variables aligned with future modeling requirements

2.2 Data Transformation Techniques

- Normalization and Standardization:
  - Apply the Power Transformer (Yeo-Johnson) to numeric variables to correct data skewness and approximate a normal distribution
  - Use StandardScaler to standardize transformed variables
  - Verify binary variables were correctly encoded as 0/1
  - Combine standardized numeric variables and binary indicators into a final modeling dataset to ensure model consistency
- Rationale for Transformations:
  - Verify the final dataset combines both standardized numeric and binary variables, optimized for use in K-Means clustering and Logistic Regression models
  - These transformations improved model performance by handling skewed distributions and ensuring all features contributed equally to distance-based algorithms

3. Analytical Methods

3.1 Software and Tools

<u>Software:</u>

Python – Used for data cleaning, preparation, analysis, and visualization

<u>Imported Libraries:</u>

Data Handling and Pre-processing:

- Pandas:
  - Function: Reading, cleaning, manipulating, and storing data
  - Usage: For reading, removing null values, sampling, and storing data
- NumPy:
  - Function: Numerical operations
  - Usage: Rounding

Data Transformation:

- Standard Scalar:
  - Function: Ensures that all features are scaled the same for Machine Learning algorithms.
  - Usage: Scaling for K-Means and Logistic Regression models
- Power Transformer:
  - Function: Ensure that all data is normally distributed for speed and efficiency of Machine Learning algorithms.
  - Usage: Normally distribute data for K-Means and Logistic Regression models
- Train Test Split:
  - Function: Splits the data into training and testing datasets
  - Usage: Test our models on untrained data

Machine Learning Algorithms:

- Knee Locator:
  - Function: Assists in determining the 'elbow point' for K-Means (optimal number of clusters) by calculating the maximum curvature of inertia vs cluster count plot.
  - Usage: Ensure the correct number of clusters are used
- KMeans:
  - Function: Segmenting data into K clusters based on distance from cluster centers determined through an iterative process of finding arithmetic means.
  - Usage: create distinguishable customer segments with unique targeting features.
- Logistic Regression:
  - Function: Performs classification through estimating dependent variables' probabilities based on relationships with independent variables.
  - Usage: To do regression models on each individual customer segment to determine which features drive classification outcomes for each segment. These outcomes can be used to target individuals within those segments.

Model Evaluation:

- Accuracy Score:
  - Function: Calculates the proportion of correct predictions within a model
  - Usage: Analyze the strength of each specific model
- Precision Score:
  - Function: Calculates the accuracy of all positive predictions
  - Usage: Analyze the strength of each specific model
- Recall Score:
  - Function: Calculates the proportion of actual positive cases that were correctly identified
  - Usage: Analyze the strength of each model
- F1 Score:
  - Function: Balance of recall and precision score
  - Usage: Analyze the strength of each model
- Confusion Matrix:
  - Function: Shows a breakdown of true positives, false negatives, etc.

    - Usage: To generate a visual of the accuracy of each regression model.
- Confusion Matrix Display:
    - Function: Allows for an easy-to-digest display of a confusion matrix.
    - Usage: To create visuals of the confusion matrix
- PCA:
    - Function: Reduces the dimensions of a model to a specified amount
    - Usage: To decompose our clusters into two dimensions so cluster separation can be viewed

Visualization:

- Matplotlib:
    - Function: Creates static visualizations for exploratory analysis or analyzing model results
    - Usage: Visualization for Elbow Method, PCA plotting, and Feature Importance Plotting
- Seaborn:
    - Function: Creates visualizations for exploratory analysis or analyzing model results
    - Usage: Confusion Matrix Plotting

3.2 Techniques Employed

- K-Means Clustering:
    - Definition: An unsupervised learning algorithm that groups data points into K distinct clusters based on standardized feature similarity, with each cluster represented by its centroid (the mean position of all data points within that cluster).
    - Justification: This method was used to segment Expedia customers into similar groups to develop understandings of different customer segments.
- Logistic Regression:
    - Definition: A supervised learning algorithm that is used to predict the probability of a binary outcome based on independent variables. It outputs values between 0 and 1 to represent the likelihood of belonging to a specific class.
    - Justification: This method was used to develop an understanding of what features are most important in influencing each individual customer group to book on Expedia.

3.3 Model Building Process

Exploratory Analysis:

- Step 1: Checking available columns and number of data points (rows)
- Step 2: Checking data types of each column
- Step 3: Check central tendency measures
- Step 3: Checking null value counts
- Step 4: Checking maximum and minimum values for each column
- Step 5: Checking for outliers (numeric, non-Boolean columns only)

Data Preparation:

- Step 1: Feature selection ~ selected based on lack of null values, outliers and significance to analysis
- Step 2: Null value removal ~ removed null values given that over 6.5 million data points did not contain null values, therefore data quality was not sacrificed
- Step 3: Extreme outlier capping ~ used capping method to reduce extreme outliers to capped values to ensure that the dataset size remains
- Step 4: Sampling ~ given the large size of the dataset, to ensure that the analysis can be properly run a sample of 1 million data points from the original dataset was taken, oversampling for the booked cases so the model has enough booked cases to train on.
- Step 5: *Recheck the sample for extreme outliers*
- Step 6: Scaling and adjusting for skewness ~ scaling the numeric columns to create a final dataset for analysis
- Step 7: Determine the correct number of clusters ~ done through elbow method and using the KneeLocator library

Model Building (K-Means):

- Step 1: Initialize scikit-learn KMeans model with 4 clusters
- Step 2: Assign cluster labels to DataFrame
- Step 3: Create 2-dimensional visual of cluster distribution
- Step 4: Create visual of the most important features in determining clusters
- Step 5: Create table of each cluster profile

Model Building (Logistic Regression):

NOTE: This was done for every individual segment using the same process

- Step 1: Initialize skit-learn Logistic Regression model
- Step 2: Split the dataset into training and testing data (test size = 30% of entire dataset)
- Step 3: Fit the regression model on the training data
- Step 4: Predict the results of the testing data
- Step 5: Create a confusion matrix
- Step 6: Retrieve accuracy measures
- Step 7: Create a regression equation
- Step 8: Add regression coefficients to coefficients DataFrame

Model Analysis:

- Step 1: Get the average precision score for all models
- Step 2: Create a visual of the precision scores for each segment's model for comparison
- Step 3: Create a combined coefficients DataFrame to analyze specific features of each segment

4. Results

4.1 Supplementary Findings

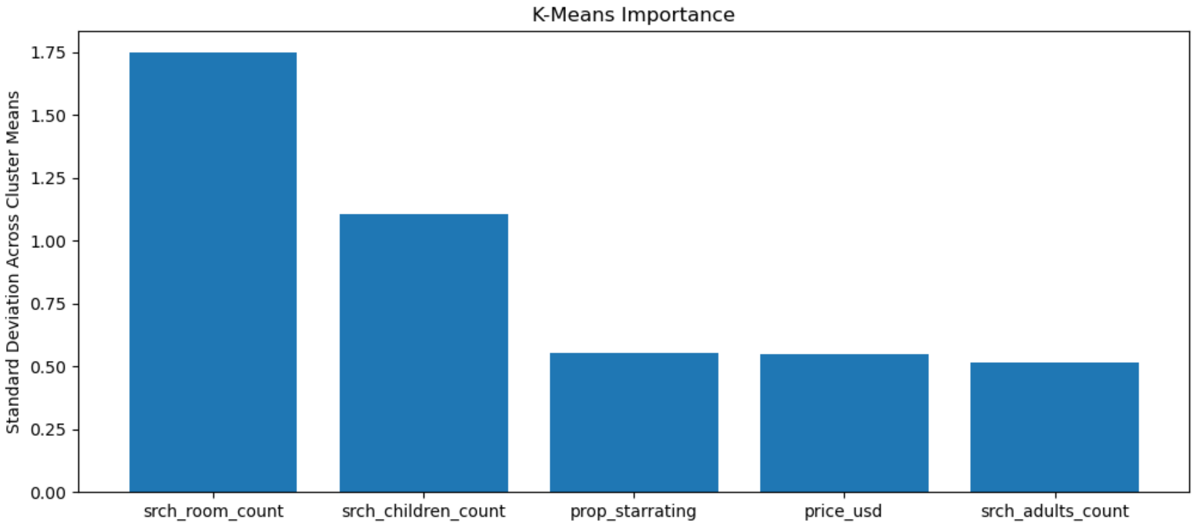


Figure 1: K-Means Feature Importance

The K-Means feature importance chart shows that room count and children count are the two strongest factors distinguishing the four customer segments. The remaining features, like star rating, price, and adults count, show moderate variation. While they do influence the segments, they are not the key drivers.

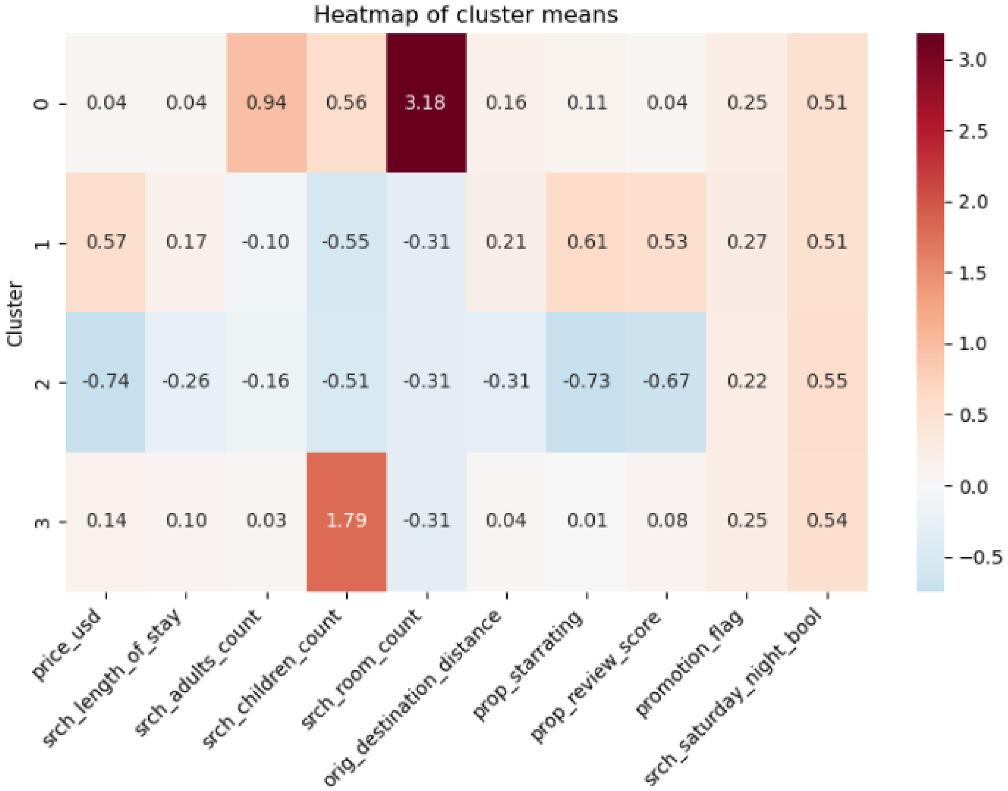
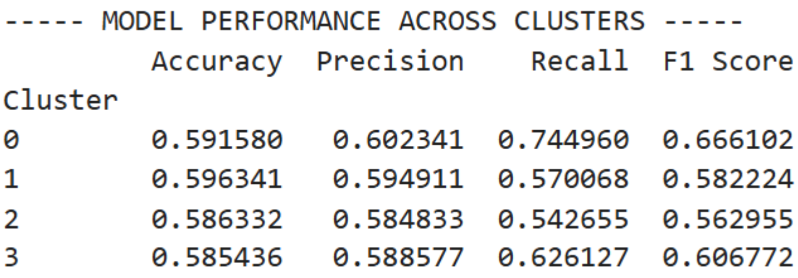


Figure 2: Heatmap of Cluster Means

The heatmap shows the key differences that define each cluster. Cluster 0 depicts extremely high room counts and adult counts, indicating large groups booking multiple rooms. Cluster 2 is the most price-sensitive group, preferring lower-rated hotels and showing below-average values across almost every feature. Cluster 3 prefers higher-priced and higher-quality properties with strong review scores and no preference to travel with children. Cluster 4 shows a strong family-orientated booking trend, with the highest children count that drives their booking likelihood.

4.2 Statistical Output



```
----- MODEL PERFORMANCE ACROSS CLUSTERS -----
         Accuracy  Precision    Recall  F1 Score
Cluster
0        0.591580   0.602341  0.744960  0.666102
1        0.596341   0.594911  0.570068  0.582224
2        0.586332   0.584833  0.542655  0.562955
3        0.585436   0.588577  0.626127  0.606772
```

Figure 3: Model Performance Across Clusters

The model performs at a similar level across all the clusters, with accuracy and precision staying around 0.58-0.60. Cluster 0 shows the strongest results with its high recall score of 0.745, making its booking identification more reliable when compared to the other clusters.

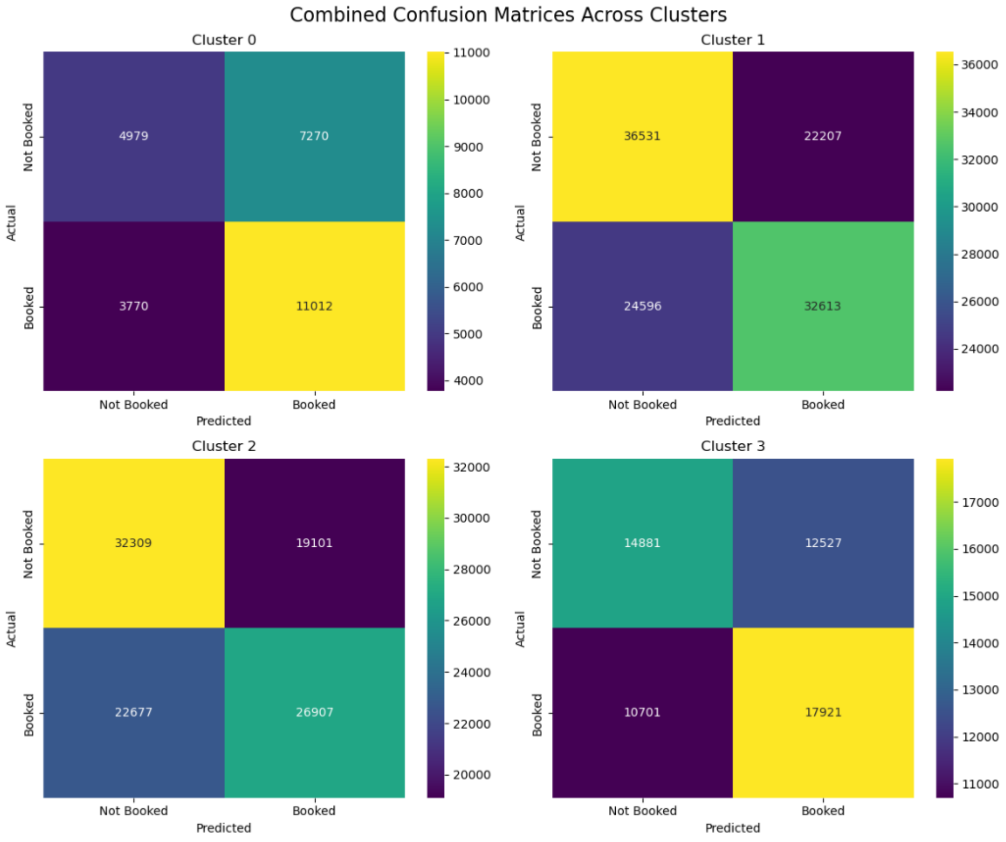


Figure 4: Confusion Matrices Across Clusters

As seen in the confusion matrices, the model performs considerably well in distinguishing booked vs. not booked. Clusters 0 and 3 show stronger booking predictions as they have higher true positives, indicating clear booking patterns among these customer groups. However, Clusters 1 and 2 is unable to fully capture the bookings well and it leads to more mixed predictions.

5. Limitations

A primary limitation is the large size of the dataset, which can slow down processing, model training, and exploratory analysis. To address this, we used a random stratified sample of 1 million search events to create a smaller yet representative subset of the data, which helped to reduce computational load while still preserving the reliability of our insights.

Another challenge is that several clusters exhibit overlapping feature patterns, which can make it difficult for the model to clearly distinguish between them and may limit classification accuracy. To address this, we conducted a deeper analysis of cluster-level feature distributions to identify more discriminative characteristics, helping improve cluster separation.

References

Data Source:

Expedia Group. (2022). *Expedia Hotel.*

Academic Reference:

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley.

Jolliffe, I. T., & Cadima, J. (2016). *Principal Component Analysis: A review and recent developments.* Philosophical Transactions of the Royal Society A.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems (NeurIPS) .

MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations* Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability

Pedregosa, F. et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830

Thorndike, R. L. (1953). *Who belongs in the family?* Psychometrika, 18(4), 267–276.

Software/Library Documentation:

Kneed: Satopaa, V., Albrecht, J., Irwin, D., & Raghavan, B. (2011). *Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior*. Kneed Documentation.

Matplotlib: Hunter, J. D., Caswell, T. A., Droettboom, M., Firing, E., & Matplotlib Contributors. (2003–2025). *Matplotlib: Visualization with Python.*

NumPy: NumPy Developers. (2011). *NumPy Example List*. SciPy.org.

Pandas: Pandas Development Team. (2019). *pandas: Python Data Analysis Library (Version 0.25) Documentation.*

Scikit-Learn: Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). *scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.

Seaborn: Waskom, M. L. (2021). *Seaborn: Statistical data visualization*. Journal of Open Source Software, 6(60), 3021.

Experiments were implemented in Python using NumPy, pandas, scikit-learn and kneed for analysis, and matplotlib and seaborn for visualization.