**Comprehensive Analysis Report on Hotel Operational Data and Clustering**

**Introduction**

This report details the design, preprocessing, and clustering analysis of a comprehensive dataset for a hotel operating in Spain. The dataset integrates customer profiles, room details, operational logs, energy consumption, and environmental factors. Our goal was to create a dataset that supports in-depth analysis of customer behavior, operational efficiency, and energy management. Various clustering algorithms were applied using PyCaret to segment the data and reveal underlying patterns.

**1. Dataset Creation Prompts**

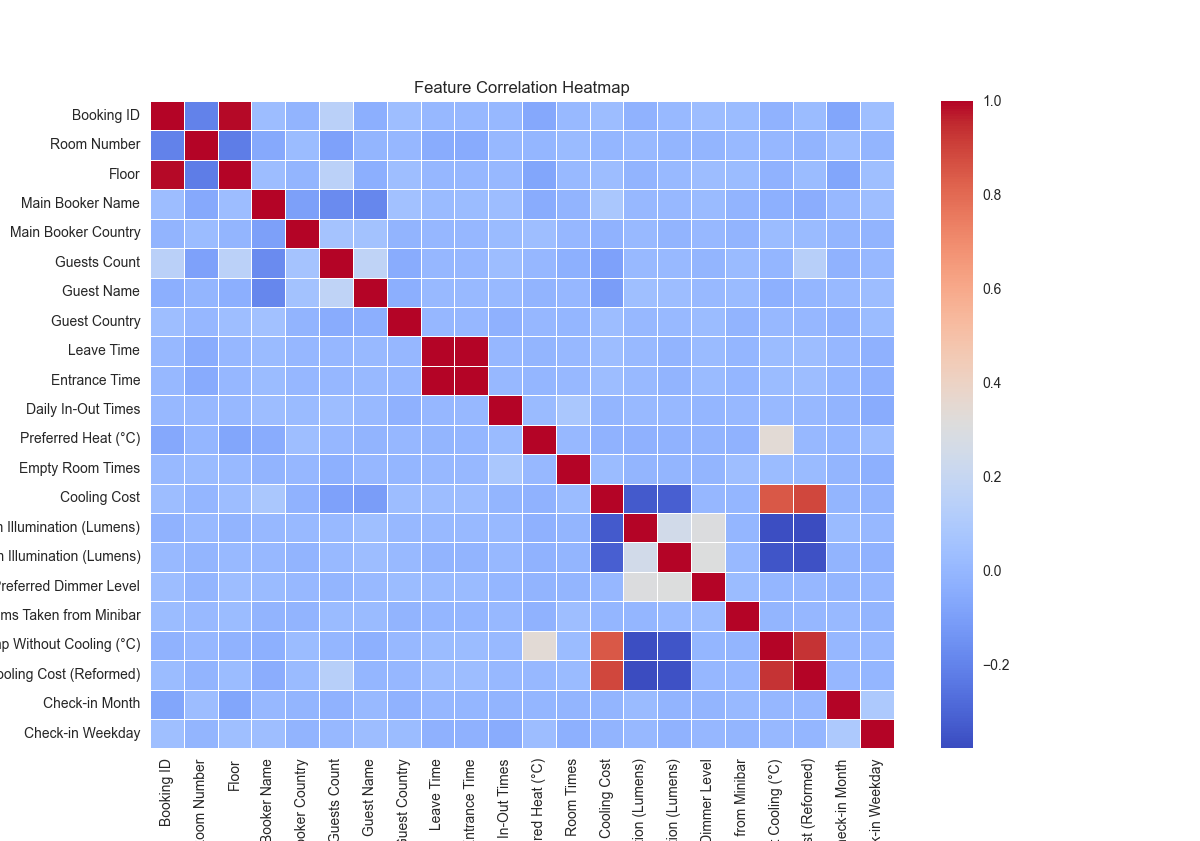
The dataset was generated based on detailed prompts to ensure it captured the multifaceted operations of the hotel. The prompts covered the following areas:

* **General Dataset Design – Extended Version**
  + **Objective:** Develop a comprehensive dataset featuring customer profiles, room details, reservation data, energy consumption, mini bar usage, lighting settings, and environmental factors.
  + **Focus:** Integrate relationships among customer behavior, room occupancy, energy usage, maintenance, customer satisfaction, local climate data, energy costs, regulations, and sustainability criteria.
* **Customer and Room Entry–Exit Information**
  + **Details:**
    - Log entry and exit times for each customer.
    - Record card numbers used for transactions, including instances with multiple guests.
    - Track relationships between different cards in the same room.
    - Capture security control logs segmented by morning, noon, and evening.
  + **Purpose:** Correlate these details with reservation cancellations, late check-ins/outs, and service quality for operational analysis.
* **Lighting and Ambient Settings**
  + **Data Points:**
    - Preferred lighting levels and specific dimmer settings for rooms and bathrooms.
    - Energy consumption of lighting systems.
  + **Objective:** Examine the connection between lighting, customer comfort, energy costs, and maintenance needs.
* **Room Orientation, Climate, and Energy Relationships**
  + **Components:**
    - Directional information for each room (north, south, east, west).
    - Local climate data including sunlight intensity, wind speed, and external temperature.
    - Analysis of how room orientation affects heating and cooling costs.
  + **Goal:** Link weather patterns and seasonal variations to energy efficiency.
* **Mini Bar Usage Data and Additional Connections**
  + **Information Captured:**
    - Daily consumption quantities for snacks and beverages.
    - Links to consumption time, customer demographics, and room conditions.
  + **Note:** While current data for mini bar usage was limited, future dataset versions will expand on these details.
* **Additional Correlations and Connections**
  + **Integrated Linkages:**
    - Relationships between customer entry/exit data, energy consumption, and security logs.
    - Associations between lighting settings, room orientation, local climate, and customer comfort.
    - Connections between mini bar consumption, demographic information, room temperature/cooling costs, and energy efficiency.
    - Dynamic links between maintenance activities and trends in customer behavior and energy performance.
  + **Outcome:** Establish multilayered connections to enable a thorough analysis of the hotel’s operational processes.

**2. Data Preprocessing**

The initial dataset was imported from an Excel file into a Pandas DataFrame and then meticulously preprocessed to prepare for clustering. The key steps included:

* **Date Processing**
  + Converted the "Check-in Date" from string format to a datetime object.
  + Extracted two new features:
    - **Check-in Month** – Numeric representation of the month.
    - **Check-in Weekday** – Numeric representation of the day of the week.
  + Removed the original date column to ensure the dataset contained only numerical inputs.
* **Handling Categorical Variables**
  + **High Cardinality (Booking ID & Room Number):**
    - Applied Label Encoding to assign a unique numeric identifier for each unique value.
  + **Low Cardinality (Room Type):**
    - Used One-Hot Encoding to create binary columns for each category, facilitating model interpretation.
* **Standardizing Numeric Features**
  + Performed standardization to adjust numeric values so that they have a mean of 0 and a standard deviation of 1.
  + **Benefit:** This normalization helps distance-based clustering algorithms perform more effectively.
* **Visualizing Feature Correlations**
  + Generated a heatmap to identify correlations among numerical features.
  + **Purpose:** Verify that the dataset was ready for clustering by ensuring feature relationships were well understood.



**3. Clustering Analysis Using PyCaret**

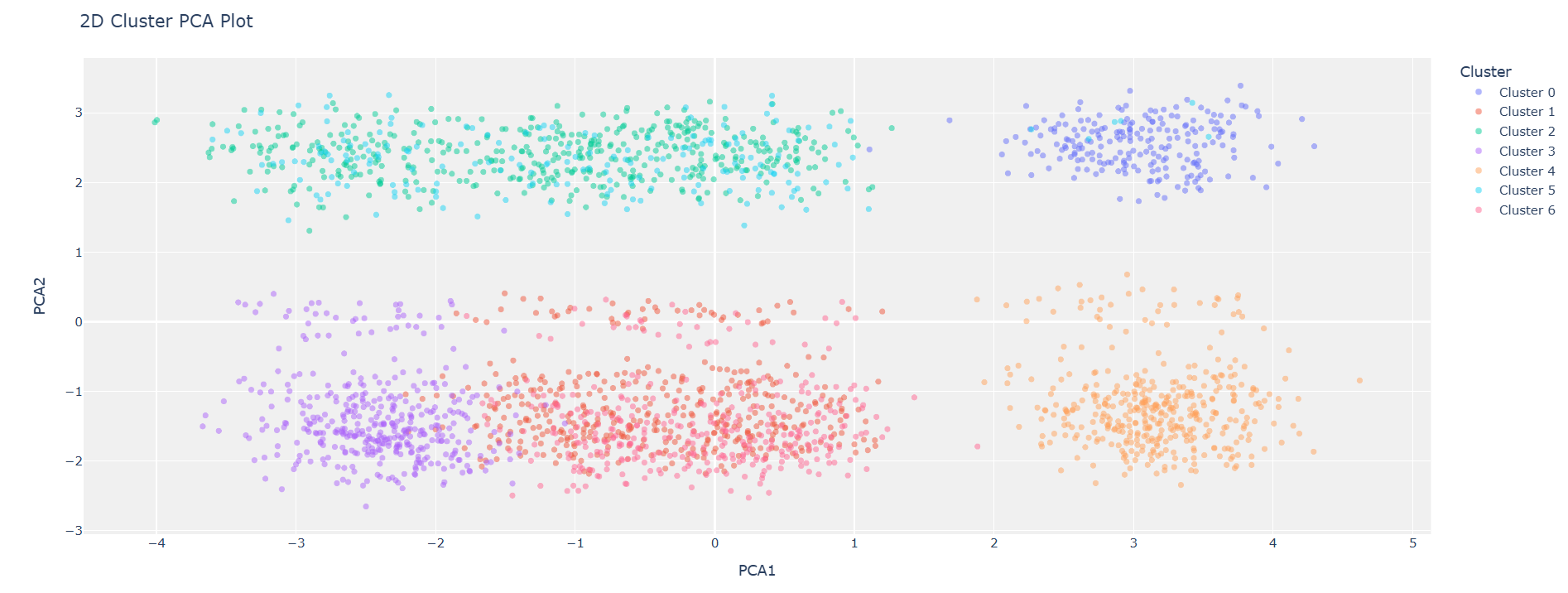
After preprocessing the dataset to ensure that all features were numerical, scaled, and free of missing values, we proceeded with clustering analysis using PyCaret. The goal was to segment the data into groups that reveal underlying patterns related to hotel operations, customer behavior, and energy usage. In this section, we detail the implementation and evaluation of three clustering methods: K-Means, DBSCAN, and Agglomerative Clustering.

**3.1 K-Means Clustering**

**Overview:**  
K-Means is a widely used clustering algorithm known for its efficiency in handling large datasets. It partitions data into a specified number of clusters by minimizing the within-cluster variance.

**Implementation Details:**

* **Parameter Setting:**
  + **Number of Clusters:** We set the number of clusters to 7. This choice was informed by initial exploratory data analysis and domain insights.
  + **Initialization:** PyCaret’s default initialization methods were used to ensure reproducibility by setting a fixed session ID.
* **Model Building:**
  + The model was created using PyCaret’s create\_model('kmeans') function, which automatically applies internal checks and provides a baseline clustering performance.
* **Evaluation:**
  + **Visualization:** A 2D PCA plot was generated using plot\_model(model, plot='cluster') to visually inspect the separation and cohesion of the clusters. The clusters were color-coded, making it easier to identify well-separated groups.
  + **Interpretation:** The PCA plot revealed clear boundaries among clusters, suggesting that the K-Means algorithm was effective in grouping similar data points. This visual clarity, combined with PyCaret’s internal metrics, confirmed that K-Means captured the underlying structure of the dataset.
* **Outcome:**
  + K-Means produced stable and interpretable clusters that aligned well with the operational characteristics and customer segmentation required for further analysis.

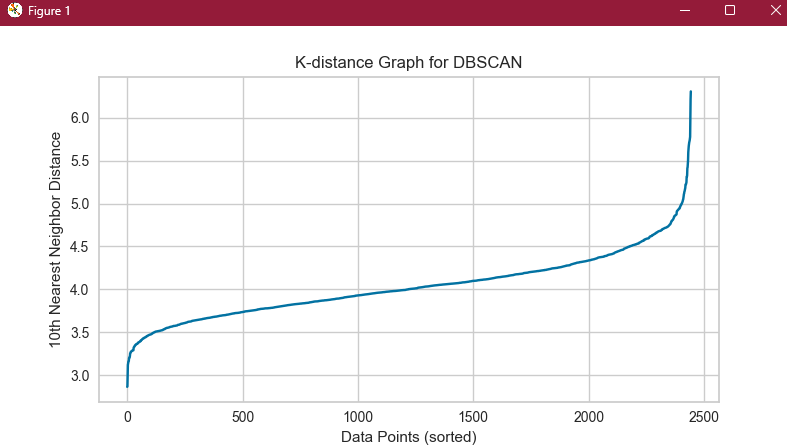


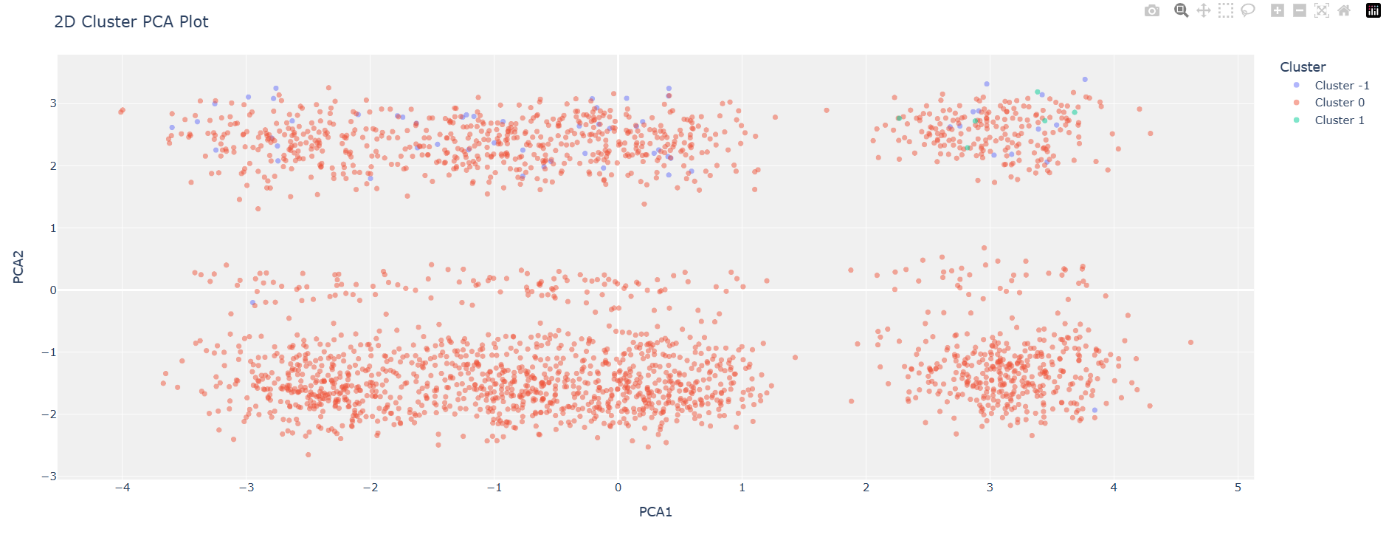
**3.2 DBSCAN Clustering**

**Overview:**  
DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm particularly useful for identifying clusters of varying shapes and sizes, as well as detecting noise or outlier points.

**Implementation Details:**

* **Parameter Selection:**
  + **k-Distance Plot:**
    - A k-distance plot was employed with k = 10 (i.e., considering the 10th nearest neighbor) to analyze the distance distribution.
    - The “elbow” observed in the plot suggested a suitable threshold for epsilon (eps), which was set at 3.8.
  + **min\_samples:** Set to 10, matching the chosen k value, ensuring that clusters would require a minimum of 10 points to be considered dense.



* **Model Building:**
  + The DBSCAN model was built using create\_model('dbscan', eps=3.8, min\_samples=10), allowing PyCaret to apply the DBSCAN algorithm based on the specified parameters.
* **Evaluation:**
  + **Cluster Characteristics:**
    - Despite the careful selection of eps and min\_samples, the model either classified too many points as noise or resulted in fragmented clusters that did not clearly represent distinct segments.
  + **Visualization:**
    - The cluster plot generated with plot\_model(model, plot='cluster') indicated a lack of cohesive groupings, with several outlier points and overly fragmented regions.
* **Outcome:**
  + DBSCAN’s sensitivity to parameter tuning proved challenging for this dataset. The results were suboptimal compared to K-Means, suggesting that while DBSCAN has advantages for certain types of data distributions, it struggled with the particular structure of our hotel operational data.
* 

**3.3 Agglomerative Clustering**

**Overview:**  
Agglomerative Clustering is a hierarchical clustering method that builds nested clusters by iteratively merging or splitting clusters based on distance criteria. It is particularly useful when the desired number of clusters is known or can be reasonably estimated.

**Implementation Details:**

* **Parameter Setting:**
  + **Number of Clusters:** The model was configured to partition the data into 4 clusters. This choice was based on preliminary analysis that suggested a natural segmentation into four distinct groups.
  + **Linkage Method:** Average linkage was chosen, which calculates the mean distance between all pairs of points across clusters. This method helps smooth out irregularities compared to single or complete linkage.
  + **Distance Metric:** Euclidean distance was used to compute the similarity between data points.
* **Model Building:**
  + The Agglomerative model was implemented using PyCaret’s create\_model('hclust', num\_clusters=4, method='average', affinity='euclidean').
* **Evaluation:**
  + **Visualization:**
    - Similar to the other methods, a cluster plot was generated using plot\_model(model, plot='cluster'). The resulting visualization allowed us to inspect the compactness and separation of the clusters.
  + **Observations:**
    - While the clusters formed were generally coherent, the fixed number of clusters sometimes forced the algorithm to split natural groupings or combine disparate data points.
    - Some disturbances in the formation of clusters were noted, likely a result of the rigid cluster count imposed by the parameter settings.
* **Outcome:**
  + Agglomerative Clustering yielded acceptable segmentation of the dataset. However, the need to predefine the number of clusters introduced some limitations, particularly when the natural data distribution did not perfectly align with the imposed structure.

**4. Final Conclusions and Recommendations**

**Summary of Findings**

* **K-Means Clustering:**
  + Delivered the best performance in terms of clear separation and stability of clusters.
  + The 7-cluster solution closely reflected the operational nuances and customer segmentation of the hotel dataset.
* **DBSCAN:**
  + Faced challenges with parameter sensitivity, resulting in either excessive noise or fragmented clusters.
  + Its performance was hindered by the data structure, making it less suitable for this specific application.
* **Agglomerative Clustering:**
  + Provided a reasonable alternative with the advantage of hierarchical insights.
  + However, the fixed cluster count sometimes led to artificial groupings that did not fully capture the natural segmentation.

**Recommendations for Future Work**

* **Refinement of DBSCAN Parameters:**
  + Additional iterative tuning or exploration of alternative density-based methods may improve its applicability.
* **Exploration of Hybrid Approaches:**
  + Combining insights from both partition-based and hierarchical methods could lead to more nuanced clustering outcomes.
* **Iterative Hyperparameter Tuning:**
  + Further experimentation with different numbers of clusters and alternative linkage criteria could help optimize the performance of Agglomerative Clustering.
* **Expansion of Dataset Dimensions:**
  + Future work should include more detailed features (e.g., comprehensive mini bar usage data) to enrich the clustering analysis and provide deeper insights into operational efficiencies.