User Review Segmentation: Unveiling Tourist Preferences

AUTHORS

Phong Ho - 1617975
R.J. Bilak - 1584507
Edmond Tuong - 1607069
Michael Zhang - 1616736

Click here for the GitHub repository or copy the link *https://github.com/whosphong/USER-REVIEW-SEGMENTATION—UAmath509.git*.

2025-09-23

Abstract

This project, User Review Segmentation: Unveiling Tourist Preferences, investigates the application of unsupervised learning techniques to analyze Google user reviews of popular tourist attractions. The objective was to identify distinct visitor segments based on rating behavior, thereby providing insights into customer sentiment and enhancing understanding of user experiences. The methodology involved exploratory data analysis and preprocessing, including the handling of missing values and standardization of ratings. Clustering was primarily performed using K-means, both on the original feature space and on data transformed via Principal Component Analysis (PCA) to facilitate dimensionality reduction and improve interpretability. Optimal cluster numbers were determined using elbow plots and silhouette scores. Additional clustering methods, including hierarchical agglomerative clustering and DBSCAN, were explored for comparative analysis. Post-clustering, groups were interpreted and labeled according to rating patterns to reveal underlying trends in user sentiment. The results identified clear visitor segments, including clusters with distinct preferences regarding accommodation, food, travel, and nature. Among the methods, K-means consistently outperformed the others, achieving the highest silhouette score (0.391), Calinski-Harabasz index (4404.151), and the lowest Davies-Bouldin index (0.818), indicating well-separated and cohesive clusters. In contrast, DBSCAN showed poor performance due to the formation of an overly dominant central cluster. These findings offer insights for companies in the tourism industry to support a more targeted marketing strategies and improvements in visitor experience.

Contents

1 Introduction

Travel and tourism is a massive global industry, generating billions of dollars annually and supporting a wide range of interconnected businesses. From airlines and hotels to restaurants, tour guides, and local shops, a single traveler's journey can impact an entire economic ecosystem. When tourism is thriving, the ripple effects are felt far beyond the travel agencies and booking platforms—small businesses grow, jobs are created, and local economies get a boost.

On the flip side, the traveler benefits too. Being able to go on vacation is often a sign of financial security and can contribute to personal well-being, relaxation, and cultural enrichment. In short, the more aligned the travel experience is with a person's interests, the more satisfying it becomes—and the more likely they are to travel again. This mutual benefit creates a strong incentive to understand what travelers actually want, so that services and experiences can be tailored to meet those expectations.

If businesses and destinations that attract tourism can most effectively understand and target potential vacationers, everyone stands to benefit. Tourists enjoy a more tailored, engaging vacation experience, while businesses and local economies see increased spending and growth. In this project, we aim to discover which methods can be used to group travelers most effectively, ultimately helping create more personalized, exciting vacations

1.1 Background Information

According to Statistics Canada, the tourism industry has generated approximately 282,987,500 jobs since 1986, underscoring its significant role in global economic growth. Countries with well-developed infrastructure and a variety of tourist attractions stand to gain even more as the travel sector continues to expand, creating more job opportunities and boosting local economies reliant on tourism.

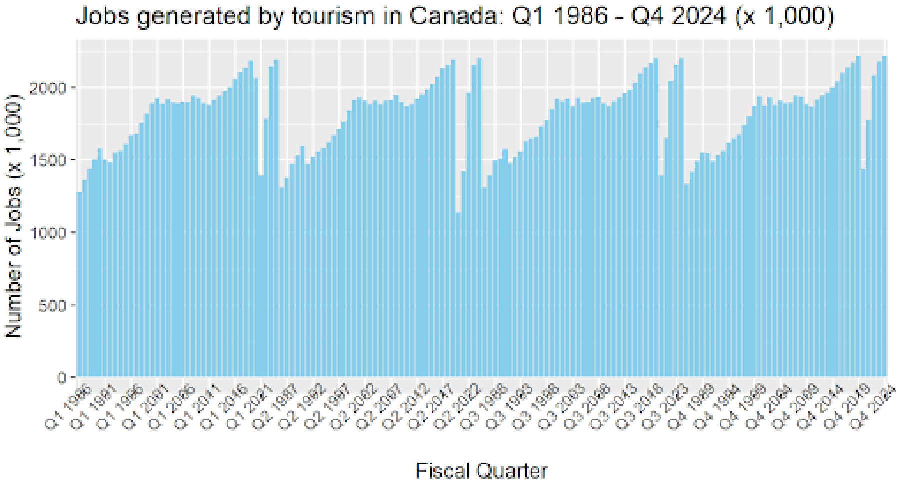


Figure 1: Jobs Generated by Tourism in Canada, as reported by Statistics Canada, 2025

Having companies that can effectively group travelers and bring them into the country can effectively help the economy of that host country.

Recent work by Ahn et al., 2020 empirically investigated how different traveler groups' experiences influence their satisfaction ratings for hotels. Their findings showed that couples typically report the highest satisfaction, while business travelers tend to be the least satisfied, with these differences attributed to varying preferences for hotel features. This insight presents companies with a valuable market opportunity: by understanding the preferences of these lower satisfaction groups, businesses can tailor their offerings to appeal to them, potentially increasing repeat visits and overall customer loyalty.

1.2 Dataset

The dataset centers on understanding how patrons perceive the quality, value, and overall experience of the services they use. This focus is particularly important, as it not only reflects levels of customer satisfaction but also uncovers how expectations and preferences vary by demographic across various types of attractions. Different categories such as museums, theme parks, zoos, historical landmarks, religious sites, and entertainment venues, may appeal to diverse demographics and fulfill different experiential needs, and user ratings offer a way to quantify and compare these subjective experiences. By analyzing the aggregated feedback, insights may be gained regarding which types of attractions resonate most positively with which individuals.

The data, sourced from the UCI Machine Learning Repository by Renjith, 2018, comprises average Google user ratings across 24 attraction categories in various European countries. Each data point represents the individual rating for all attractions within a given category, using a 1 to 5 scale, where 1 denotes a poor experience and 5 denotes an excellent experience. Any review of a zero was interpreted as the individual not leaving any reviews in that category.

Averaging by attraction type helps facilitate the identification of broader patterns in user sentiment, making it possible to assess a more general public perception and satisfaction levels for different attraction types. Ultimately, the project seeks to interpret these patterns not only to understand how certain attractions are received by certain like groups, but also to offer actionable insights for tourism authorities, city planners, and business owners aiming to enhance visitor experience, allocate resources more effectively, and tailor services to better meet public expectations.

1.3 Research Questions

The primary goal of this paper is to address the objective of:

"*Comparing different clustering methods and evaluate their effectiveness in grouping individuals based on their attraction preferences.*"

We also pose the following mathematical questions to gain a deeper understanding of the methods we used.

- How do the results from each clustering method compare when applied to the user review data?
- Do all methods produce similar cluster groupings, or do they reveal different patterns in the data?
- Does the optimal number of clusters differ between each method of clustering?
- Are the resulting clusters balanced, or do some methods create a few large clusters and many smaller ones?
- How do each of the methods handle outliers and noise in the data?

2 Mathematical Formulation

Throughout our analysis we will compare various clustering algorithms and dimension reduction methods. It is crucial to know what is the mathematical motivation behind how they work such as the theoretical and formulation of the models. In total there are a total of 3 clustering algorithms that we wish to compare their analysis. Elaborating on them we have:

2.1 Clustering Algorithm

2.1.1 Agglomerative Clustering

Agglomerative clustering starts with every point in its own cluster, then merges the pair (A,B) of clusters with the lowest set-wise distance or linkage until left with k clusters. The choice of linkage is very important. Pedregosa et al., 2011 includes these linkages:

- **Single-linkage** is the fastest one and equals:

$$\min_{a\in A, b\in B} d(a,b).$$

- **Complete-linkage** is comparable, and equals:

$$\max_{a\in A, b\in B} d(a,b).$$

- **Unweighted average** is the distance between the average over all a and all of b.
- **Weighted average** starts by merging clusters via some distance metric, $d(a,b)$, then when A and C are merged, the new cluster's distance to cluster B is:

$$\frac{d(a,b)+d(c,b)}{2}.$$

Initially, it does not look weighted, but overall a point has less impact on the distances of a cluster it is in if it has been a part of more merges.

- **Centroid linkage** is the distance between the centroids, which can be written as:

$$D\left(C_A, C_B\right) = \left\| \frac{1}{|C_A|} \sum_{x \in C_A} x - \frac{1}{|C_B|} \sum_{x \in C_B} x \right\|$$

- **Median linkage** is like centroid linkage, except when A and C are merged, the new centroid is the median of the two previous centroids. It is to centroid linkage as weighted average is to unweighted average. For both centroid and median, the lowest linkage is not guaranteed to decrease as clusters get merged, which complicates visualization.

- **Ward linkage** is the variance of the new cluster, minus the variances of the old clusters. This is very natural if the data is assumed to come from a Gaussian mixture model (i.e., a set of normal distributions).

To determine the most suitable linkage for hierarchical clustering, we can refer to the Cophenetic Correlation Coefficient (CPCC). The CPCC measures the correlation between the original pairwise dissimilarities (such as Euclidean distance or other distance metrics) and the cophenetic distances derived from the hierarchical clustering tree. The cophenetic distance represents the height at which two data points or objects are first merged into a cluster within the dendrogram.

The CPCC quantifies how well the dendrogram preserves the original pairwise distances between data points. It is calculated as follows: Let d_{ij} denote the Euclidean distance between the i-th and j-th points, and let t_{ij} represent the dendrogram distance, which is the height at which the objects are first merged. The CPCC is computed using the formula from Kumar and Toshniwal, 2016:

$$\text{CPCC} = \frac{\sum_{i<j} \left(d_{ij} - d'_{ij}\right) \cdot \left(t_{ij} - t'_{ij}\right)}{\sqrt{\left(\sum_{i<j} \left(d_{ij} - d'_{ij}\right)^2\right) \cdot \left(\sum_{i<j} \left(t_{ij} - t'_{ij}\right)^2\right)}}$$

In this equation, d'_{ij} and t'_{ij} represent the means of the original pairwise dissimilarities and the cophenetic distances, respectively. A higher CPCC value, close to 1, indicates that the dendrogram better preserves the original pairwise distances, suggesting that the clustering method has effectively captured the true relationships among the data points.

To predict with agglomerative clustering, we could add each new point as a new cluster and continue the algorithm, disallowing merges between old clusters.

2.1.2 K-Means Clustering

K-Means refers to a variety of algorithms to find k clusters, with every point being closer to its cluster's centroid (mean) than to any other mean. The most common is Lloyd's algorithm. It begins by randomly selecting k points to act as the initial cluster center. Then, each data point in the dataset is assigned to the nearest of these means based on some distance metric, typically Euclidean distance. After all points have been assigned, the algorithm updates each cluster center by computing the centroid of all the points currently assigned to that cluster. This is repeated until convergence. This algorithm is fast, but is only guaranteed to converge for Euclidean distance, and even then often converges to a "poor" result. Additionally, unlike agglomerative clustering, a change in k produces drastically different results, and this parameter needs to be tuned. Therefore, the entire algorithm must be run many times for good results.

Our version of prediction in the K-Means setting is to simply add new points to the cluster of the closest mean, which is much simpler than our other notions of prediction.

2.1.3 DBSCAN Clustering

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) views a cluster as a high density region surrounded by low density "noise". Points with more than n points within a distance of ε are considered core points and form a cluster with other nearby core points; non-core points within ε of a core point are border points of that cluster; and all other points are noise. The main parameter that requires particular attention to tune is ε (the model fails completely if not), and n can be adjusted intuitively to keep the proportion of noise at a reasonable level. Unlike the other two, the number of clusters naturally follows from the choice of ε and n.

DBSCAN prediction entails checking for core points within ε of the new points, and if so putting the new point in the cluster of a randomly selected nearby core point.

2.2 Dimensionality Reduction for Data Visualization

Dimensionality reduction plays a crucial role in helping us visualize our data, particularly when dealing with a high number of features, as in the case of the 24 features in our analysis. As we will see later in the mathematical analysis, these features can be reduced to 6 broad categories based on their natural groupings.

Let X be the design matrix, where the i-th row is denoted by $X^{(i)}$, representing the i-th of n points in the data, with components $x_j^{(i)}$. Assume that the mean of $x^{(i)}$ is 0.

2.2.1 Principal Component Analysis (PCA)

PCA finds the lower dimensional subspace to project the data onto, so that the projected data keeps (or explains) the most variance. For 1 dimension, if v is a unit vector, the variance is

$$\sum_{i=1}^{n} \frac{(x^{(i)} \cdot v) \cdot |v|^2}{v \cdot v} = v^\top X^\top X v.$$

such can be maximized by the method of Lagrange Multipliers, and will yield a maximum when

$$2X^\top Xv = 2\lambda v,$$

i.e. when v is an eigenvector of $X^\top X$ (i.e. the right singular vectors of X), and λ is the corresponding eigenvalue. The variance in this direction is then:

$$v^\top \lambda v = \lambda.$$

Therefore, our subspace is the span of eigenvectors of $X^\top X$, with preference to higher eigenvalues.

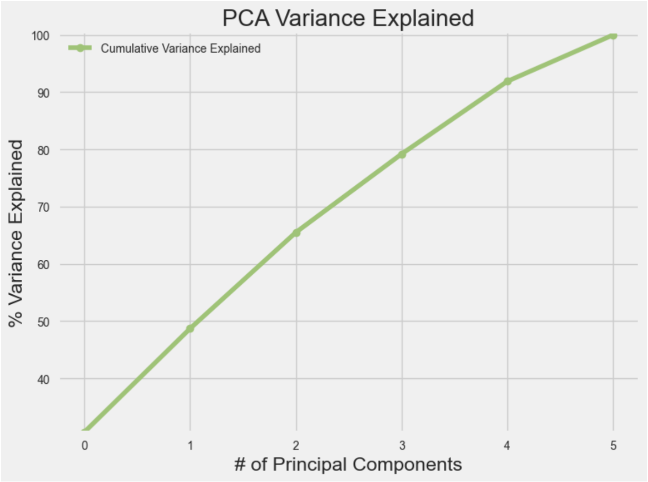


Figure 2: Variance Explained plot of each component on the 6 categories

Figure 2 illustrates the amount of variance explained by each principal component, ordered by eigenvalue. The eigenvalues for the first five components are as follows:

$$[1.83851435, 1.08723072, 1.00676677, 0.8226788, 0.76023398],$$

which were computed via Singular Value Decomposition (SVD) as explained.

It is worth considering that PCA, being linear, gives high influence to faraway points; this is likely not a concern for us since the data has known bounds. If there is structure to the data but it is not linear, PCA will not enable it to be detected. Lastly, also note that faraway points may be projected to the same point, so clusters within PCA may not translate well back to real data.

2.2.2 Multidimensional Scaling (MDS)

MDS refers to a variety of methods based on minimizing a loss function (or "strain"), such as:

$$\sum_{i,j=1}^{n} \left(\left| \mathbf{x}^{(i)} - \mathbf{x}^{(j)} \right|^p - \left| s^{(i)} - s^{(j)} \right|^p \right)^2,$$

over the s_i, which are lower-dimensional vectors representing the data points $\mathbf{x}^{(i)}$. In words, MDS aims to preserve the pairwise distances between points rather than focusing on the variance of the data. For instance, with $p = 2$, we can write:

$$s^{(i)}s^{(j)} = -0.5\left(\left|s^{(i)} - s^{(j)}\right|^2 - \left|s^{(i)}\right|^2 - \left|s^{(j)}\right|^2\right),$$

to minimize a somewhat similar function:

$$\sum_{i,j=1}^{n}\left(\mathbf{x}^{(i)}\cdot\mathbf{x}^{(j)} - s^{(i)}\cdot s^{(j)}\right)^2 = \operatorname{tr}\left(A^\top A\right),$$

where $A := X^\top X - S^\top S$. Clearly, this function is minimized when $X^\top X = S^\top S$, so, in the same spirit as PCA, dropping the components of X with low singular values should produce a good approximation.

Further generalizations (including non-metric MDS, which works even on non-Euclidean distance, which travel ratings may fall under) are not conducive to similar analysis, instead relying on the iterative SMACOF algorithm.

MDS is limited by the fact that its dimensions generally do not lead to easy interpretation; if the data had labels, the plot could give insight into the broad shape of the existing categories. Therefore the hope is that its clusters more accurately reflect the original data than PCA.

2.2.3 Isometric Mapping (Isomap)

Isomap is a variant of MDS where the distance metric is defined as the length of the shortest path between two points, consisting of segments only from a point to one of its nearby points. "Nearby points" can either be those within a certain radius r, or the k nearest neighbors of a point. In other words, distance is measured only along the dense regions of the data. While Isomap shares the same caveats as MDS, it is slower due to the more complicated metric and adds an additional parameter to tune.

If the dataset has been "folded" in such a way as to bring points that are far away in some confounding variable closer together (e.g. extremely rich but frugal people having some of the same travel preferences as middle class people), Isomap can be useful to unfold the dataset. It shares this advantage with DBSCAN since both are nearest neighbor methods, so we expect the two methods to be highly compatible.

2.3 Clustering Evaluation Metrics

A natural question that arises is: how does one decide how many components are optimal for the analysis? While it might seem intuitive that more components or dimensions would always result in a better representation of the data, this is not necessarily the case. In practice, 2-3 dimension are often chosen to strike a balance between dimensionality reduction and interpretability, making it easier to visualize and analyze the data. However, it's crucial to compare the performance of these components using clustering metrics.

In this case, we employed several clustering evaluation metrics to guide our decision-making: Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index specifically for K-means and Agglomerative since these methods allow us to decide the number of clusters to have. These are the only metrics available on scikit-learn that work even without ground truth labels. DBSCAN is excluded from the Davies-Bouldin Index due to its ability to determine the cluster based on various parameters. Also note that average linkage was used for Agglomerative method. These metrics assess the quality of the clusters based on different criteria:

- **Silhouette Score**: The silhouette score for a single point is given by:

  $$\frac{b-a}{\max(a,b)},$$

  where a is the mean distance to other points in the same cluster, and b is the mean distance to points in the nearest neighboring cluster. A silhouette score above about 0.6 indicates a good clustering, while a negative value may indicate that a point belongs to a wide cluster that is close to a more compact cluster. The silhouette score for the entire set of clusters is the average silhouette score for all points.

- **Calinski-Harabasz Index**: This index is the F-test statistic applied to clustering, which is the ratio of between-group variance to within-group variance. A higher Calinski-Harabasz (CH) index indicates better clustering results. If the clusters overlap or are nested, the between-group variance will be relatively low, resulting in a lower CH index. Although this index can be helpful, it should not be used for hypothesis testing as its values tend to be much higher than those of a typical F statistic.

- **Davies-Bouldin Index**: The Davies-Bouldin (DB) index for cluster C_i is defined as the maximum similarity between cluster C_i and all other clusters, given by:

  $$\mathrm{R}_{ij} = \frac{s_i + s_j}{|\mu_i - \mu_j|},$$

  where μ_i is the centroid of cluster C_i, and s_i is the average distance from points in C_i to the centroid μ_i. The DB index is minimized when the clusters are well-separated and compact. The overall DB index is the mean of the individual DB indices for all clusters. Lower DB index is better.

3 Mathematical Analysis

We begin with the exploratory data analysis (EDA) and preprocessing of the data that was provided. The original dataset showed signs of errors, particularly in the covariates, which indicated that there were 26 variables. However, upon investigation of the data and the code provided using the pandas package in Python, it was found that 24 of the 26 variables were the expected categories, 1 was the user ID that reviewed the attraction, and the final variable was an unnamed column that needed to be removed.

As part of preprocessing, we also relabeled columns to make them more accessible for reference during the analysis. The data showed signs of duplicate records, specifically the pairs (2,4), (670,674), and (1338,1346), which were removed. There could be an argument for keeping these pairs in cases where users might be considered "twins" (i.e., two users leaving identical or near-identical reviews), but the decision was made to remove them to avoid skewing the analysis.

We also addressed missing and corrupted values for users 1347, 1348 and 2712, where there were missing data in the "garden" and "pizza and burger" categories, respectively. These missing values were imputed with 0s since 0 indicates that the user did not review those particular attractions, as previously mentioned in 1.2).

Moving forward, we considered outliers. Since Google reviews are restricted to a 1-5 scale, outliers within this range may not necessarily be errors and are just extreme values within the allowed range. These extreme values (such as consistently giving very high or very low ratings) could reflect legitimate user experiences. Removing them might lead to an inaccurate representation of the general population. Therefore, it was crucial to keep all review values, as they are meaningful to our conclusions. However, there were indications of bimodal distributions for certain attractions, which we took note of but chose not to address directly in the analysis.

In the variable selection phase, we focused on the importance of understanding the relationship between variables, as one might explain the other. This can be assessed by using a correlation heatmap. In practice, values greater than a threshold of 0.7-0.8 in correlation indicate that one of those variables should be removed to avoid multicollinearity. The largest correlation we found was 0.62 between "theatres" and "parks," which is not large enough to warrant concern. Therefore, both variables were retained for further analysis.

To better facilitate our analysis, we grouped the individual features into a broader subcategories based on their nature and relevance. This categorization helped us streamline the analysis and interpret the data more effectively. These categories are:

- **Entertainment:** Features: Theatres, Dance Clubs, Malls
- **Food & Travel:** Features: Restaurants, Pubs/Bars, Burger/Pizza Shops, Juice Bars, Bakeries, Cafes
- **Places of Stay:** Features: Hotels, Resorts, Other Lodgings
- **Historical:** Features: Churches, Museums, Art Galleries, Monuments
- **Nature:** Features: Beaches, Parks, Zoos, View Points, Gardens
- **Services:** Features: Local Services, Swimming Pools, Gyms, Beauty Spas

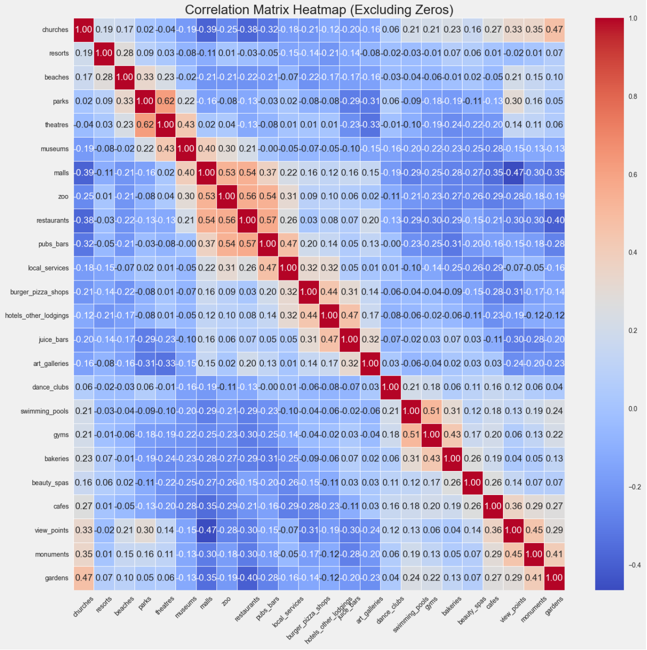


Figure 3: Heatmap of correlation matrix pairwise between all 24 categories

We then moved into applying PCA (Principal Component Analysis) to our data as described in 2.2.1, we compared the metric scores of various clustering algorithms after applying PCA using different numbers of components (ranging from 2 to 5). In **Figure 4**, it is clear that the configuration with 2 principal components (represented by the green and blue curves) yields the most favorable clustering results, outperforming the other configurations with more components. Notably, the two components retained 66% of the variance in PCA, which we found to be sufficient for effective clustering. This supports the decision to use 2 components as the optimal choice for our clustering analysis.

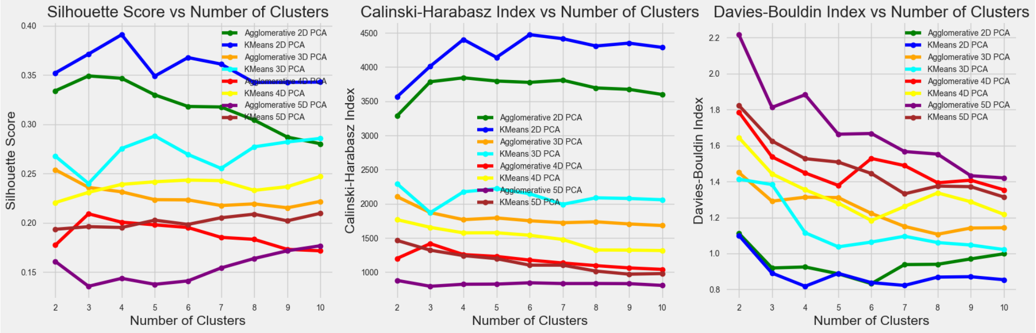


Figure 4: Comparison of the effect of varying the number of principal components in PCA, evaluated using Silhouette, Davies-Bouldin, and Calinski-Harabasz

Similarly, **Figure 5** shows the same plot but using the MDS dimension reduction method.

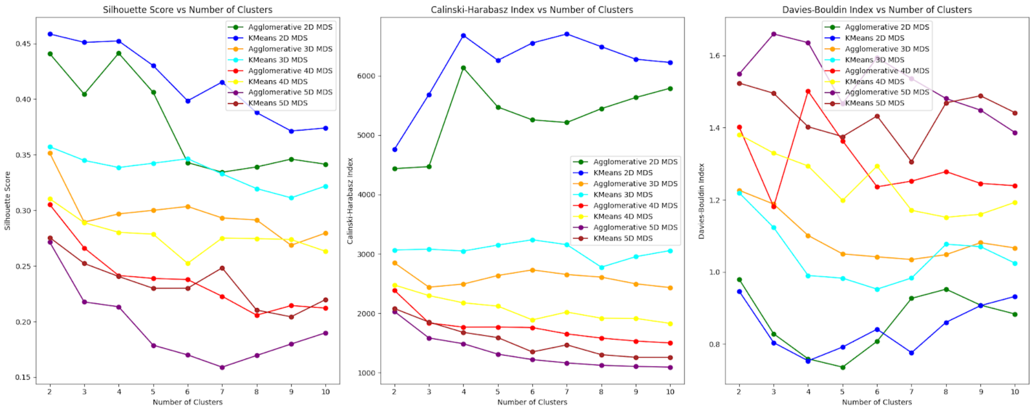


Figure 5: Comparison of the effect of varying the number of MDS components, evaluated using Silhouette, Davies-Bouldin, and Calinski-Harabasz

To explore groups of people based on these two principal components, we proceeded with Agglomerative Clustering. This clustering method requires a predefined linkage criterion, and to evaluate the effectiveness of our chosen linkage, we used the Cophenetic Correlation Coefficient (CCC). **Figure 6** shows the result for the Cophenetic Correlation Coefficient of different linkage based on the 2 principal components.

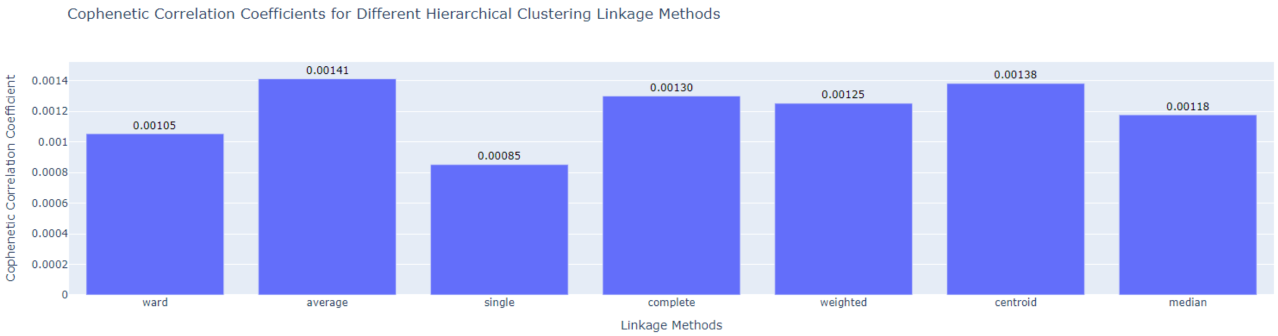


Figure 6: Cophenetic Correlation Coefficients for various linkage types in agglomerative clustering, applied to the first two principal components of PCA on the user review dataset.

The average linkage method, which was identified as the largest value in our plot, is the preferred choice in our case. While Agglomerative Clustering is deterministic, it still allows us to choose the number of clusters based on a distance cutoff from the dendrogram. However, we can also use clustering metrics such as the Silhouette score and the Calinski-Harabasz index to guide this decision. Referring back to the **Figure 4**, we observe that with 2D PCA, Agglomerative Clustering with 2 clusters and average linkage performs the best in both the silhouette score and Calinski-Harabasz index. Given that it excels in two key metrics, we favor this configuration of 2 clusters for our analysis.

Similarly to Agglomerative Clustering, K-Means allows us to pick the numbers of clusters however it is required us to predetermine the number of clusters before performing any calculations. In contrast to Agglomerative, where the number of clusters is determined after examining the dendrogram, K-Means requires us to specify the cluster count upfront. One common method for selecting the optimal number of clusters is the elbow method, which is based on examining the inertia which is a measure of how internally consistent the clusters are.

When applying the elbow method on the 2D PCA data, we plot the inertia values against the number of clusters to identify the "elbow" point, where the decrease in inertia starts to slow down. This elbow indicates a good trade-off between the number of clusters and the variance explained by them. The plot of inertia against the number of clusters is shown in **Figure 7**.

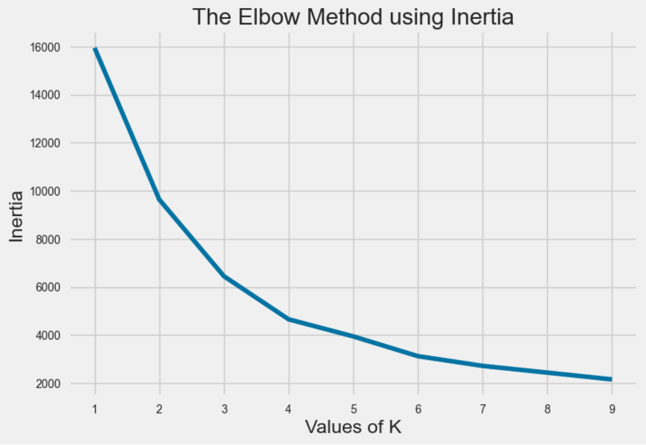


Figure 7: Intertia plot for various cluster numbers of K-Means

The point of the elbow method is to observe where the curve begins to flatten out. Although this is inherently subjective, we chose 4 clusters as the optimal number. To validate our choice, we can refer back to the **Figure 4** . Looking at the blue curve across all metrics, we see that 4 clusters perform the best in terms of the silhouette score and Davies-Bouldin index. Since it performs best in 2 out of 3 metrics, while the other configurations perform less favorably, we can confidently stick with 4 clusters for our final model.

Finally, we move on to DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Unlike the previous two methods, which allow us to choose the number of clusters either before or after applying the model, DBSCAN does not explicitly require us to specify the number of clusters. Instead, DBSCAN forms clusters based on the density of points within a specified neighborhood, controlled by the parameter epsilon (ϵ).

While DBSCAN does not have a fixed number of clusters, we can still control it through the epsilon parameter, which defines the radius of a neighborhood around each point. Additionally, methods like the elbow method, used for determining the K-distance like the one shown in **Figure 8**, can help us determine an optimal epsilon value. This K-distance plot allows us to visually identify a point at which the distance between points starts to increase significantly, which can guide the selection of epsilon.

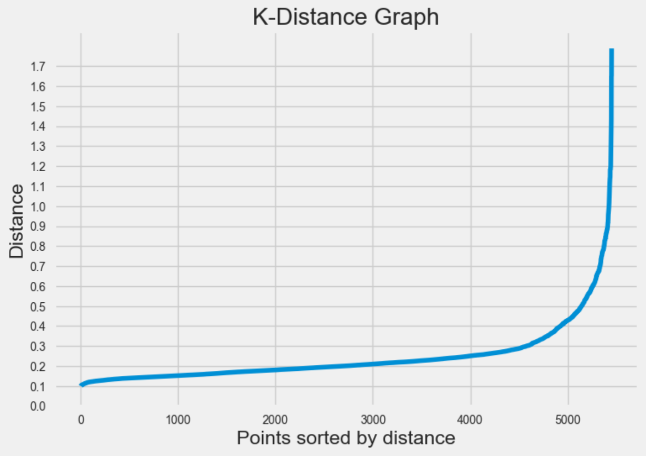


Figure 8: K-Distance plot for determining the epsilon value using the elbow method for DBSCAN.

Similar to inertia, we can look for an elbow point in the K-distance plot. However, this approach is subjective, and in our case, we identified the elbow around an epsilon value of 0.3. That said, the K-distance graph does not explicitly tell us which epsilon values result in meaningful clusters. For instance, certain epsilon values might lead to a scenario where DBSCAN forms one large cluster after subtracting the noise, which does not offer a meaningful clustering solution. This can happen if the dataset is sparse or if DBSCAN does not find enough points with the given epsilon to form valid clusters due to too many noise points or insufficient density. In such cases, the algorithm may fail to identify clusters with more than one unique label.

To address this, we focused on evaluating the quality of clusters for different epsilon values. We iterated through epsilon values from 0.1 to 1 in increments of 0.05, assessing the clustering results based on meaningful metrics. After evaluating the performance across multiple epsilon values, we found that an epsilon of 0.2 yielded the best results, as it performed the best in both the Silhouette Score and Calinski-Harabasz Index. This choice was further supported, as it outperformed the other values in 2 out of 3 metrics, making it the optimal choice for our DBSCAN clustering analysis.

Looking at **Figure 9**, we found that with an epsilon value of 0.2, DBSCAN identified 4 true clusters and 1 noise cluster. This result aligns with the performance of DBSCAN for this particular epsilon value, as it effectively separated the data into meaningful clusters, while the noise cluster represented points that did not fit well into any of the identified clusters. This outcome was supported by the higher Silhouette Score and Calinski-Harabasz Index for epsilon = 0.2, indicating that the clustering structure was both well-defined and meaningful. The noise cluster, although present, did not significantly affect the overall clustering quality.

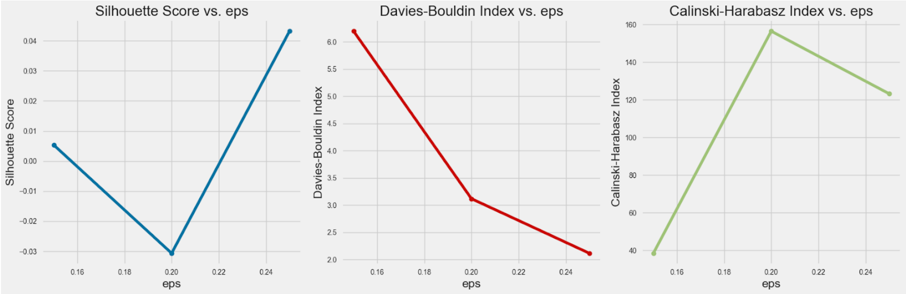


Figure 9: Comparison of evaluation metrics for various epsilon values for DBSCAN

4 Results

4.1 PCA

The results of each method are as follows: Agglomerative, K-means, and DBSCAN resulted in 3, 4, and 4 clusters, respectively. Since we applied PCA before clustering, we were able to visually assess how the data is grouped in the reduced 2D space. **Figure 10** below illustrates this by plotting each data point along the axes of Component 1 and Component 2, with points color-coded according to their respective clusters. This visual representation allows us to compare how the clusters are distributed across the three methods.

For PCA: In Agglomerative and K-means clustering, we observe that the clusters are not very distinct. This is evident from the noticeable overlap between clusters, which suggests that the algorithms are not capturing the underlying structure of the data very well. This observation aligns with the silhouette score, which we will discuss later. When clusters overlap significantly, it typically indicates poor separation, meaning the algorithm has trouble distinguishing between the sub-categories. Ideally, a good clustering algorithm should show clear separations between clusters that are often easily distinguishable by humans in a scatter plot.

DBSCAN, on the other hand, shows a very skewed distribution in its clusters, particularly in Cluster 0, which contains the majority of the data points. The remaining clusters have very few members, if any. This distribution suggests that DBSCAN is less meaningful for our dataset, as it tends to group most of the points into a single large cluster while leaving only sparse and scattered points in the others. Given this behavior, DBSCAN does not appear to capture the ideal groupings in this particular case.

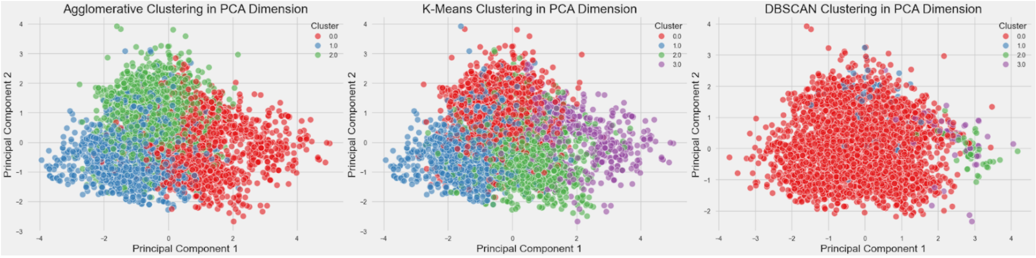


Figure 10: PCA plots for the different clustering algorithm

4.2 MDS and Isomap

For classic MDS (**Figure 11**), the data is quite globular with visible breaks, which seem to be best captured by agglomerative clustering. DBSCAN has approximately the same 2 big clusters but again creates a lot of small clusters. K-Means consistently produces this result where the centroids form a sort of rhombus but the clusters do not seem particularly well-separated.

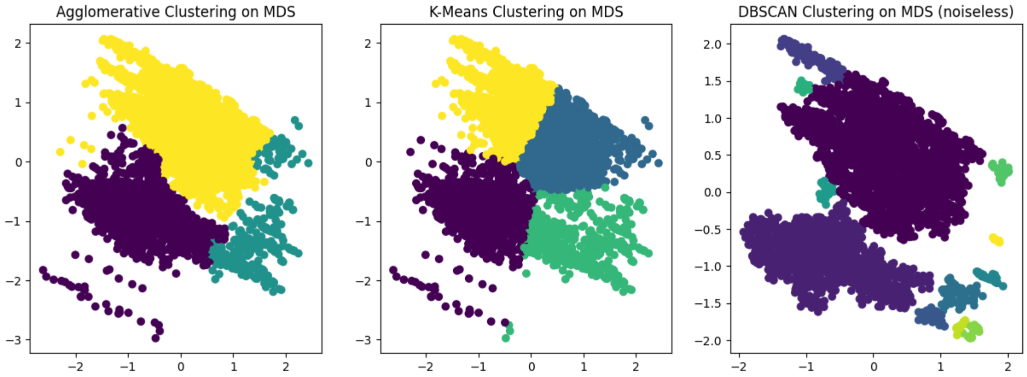


Figure 11: MDS plots for the different clustering algorithm

Non-metric MDS (**Figure 12**) produces points with no visually discernible structure other than being square. Here it is most obvious that K-means tends to prefer equal size clusters, and DBSCAN generates a very large one that envelops several smaller ones. For agglomerative clustering, the Davies-Bouldin score stays at a constant 271.57 no matter which linkage or cluster number is used, which is absurdly high and might indicate some sort of error. This seems quite bad as a vehicle for clustering.

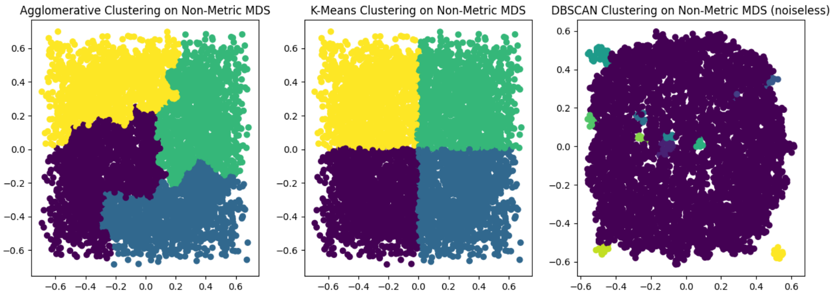


Figure 12: Non-Metric MDS plots for the different clustering algorithm

20 neighbors in Isomap (**Figure 13**) causes very similar results to MDS (tilt perspective 90 degrees counterclockwise). This indicates that this number of neighbors is not small enough to separate the method from Euclidean distance. The main difference is that agglomerative clustering now sees 2 clusters as slightly better.

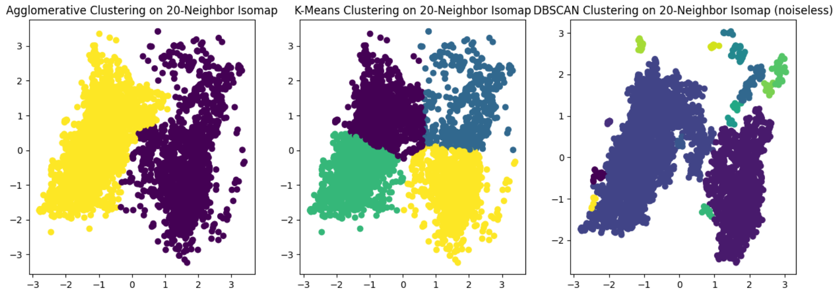


Figure 13: 20-Neighbour Isomap plots for the different clustering algorithm

In 5 neighbor isomap (**Figure 14**), some contingent near the top of the graph appears to be separating from the rest of the points (as compared to 20 neighbor isomap), which is captured by all 3 methods, though DBSCAN seems to see a lot of it as noise.

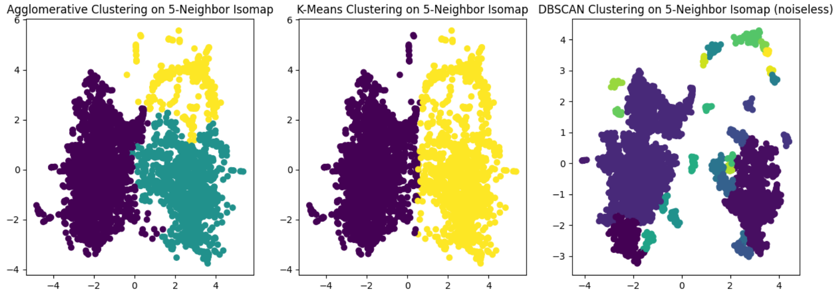


Figure 14: 5-Neighbour Isomap plots for the different clustering algorithm

4.3 PCA Result Analysis

For PCA, we continue by delving into the statistical analysis by examining the median ratings of various subcategories for each cluster. This helps us gain a deeper understanding of the types of individuals that fall within each group, as shown in the **Figure 15**.

Starting with the Agglomerative method, we observe that Cluster 0 represents individuals who are more neutral about all attractions. These individuals are not overly excited nor particularly disappointed with their experiences; they appear to be generally satisfied with each attraction. The key distinction between Clusters 1 and 2 lies in their ratings for the place of stay. Cluster 1 shows clear dissatisfaction with their accommodation, with a median rating of approximately 1.5, whereas Cluster 2 has a significantly higher rating of about 3.4. This difference in accommodation satisfaction is the defining characteristic that sets these two clusters apart.

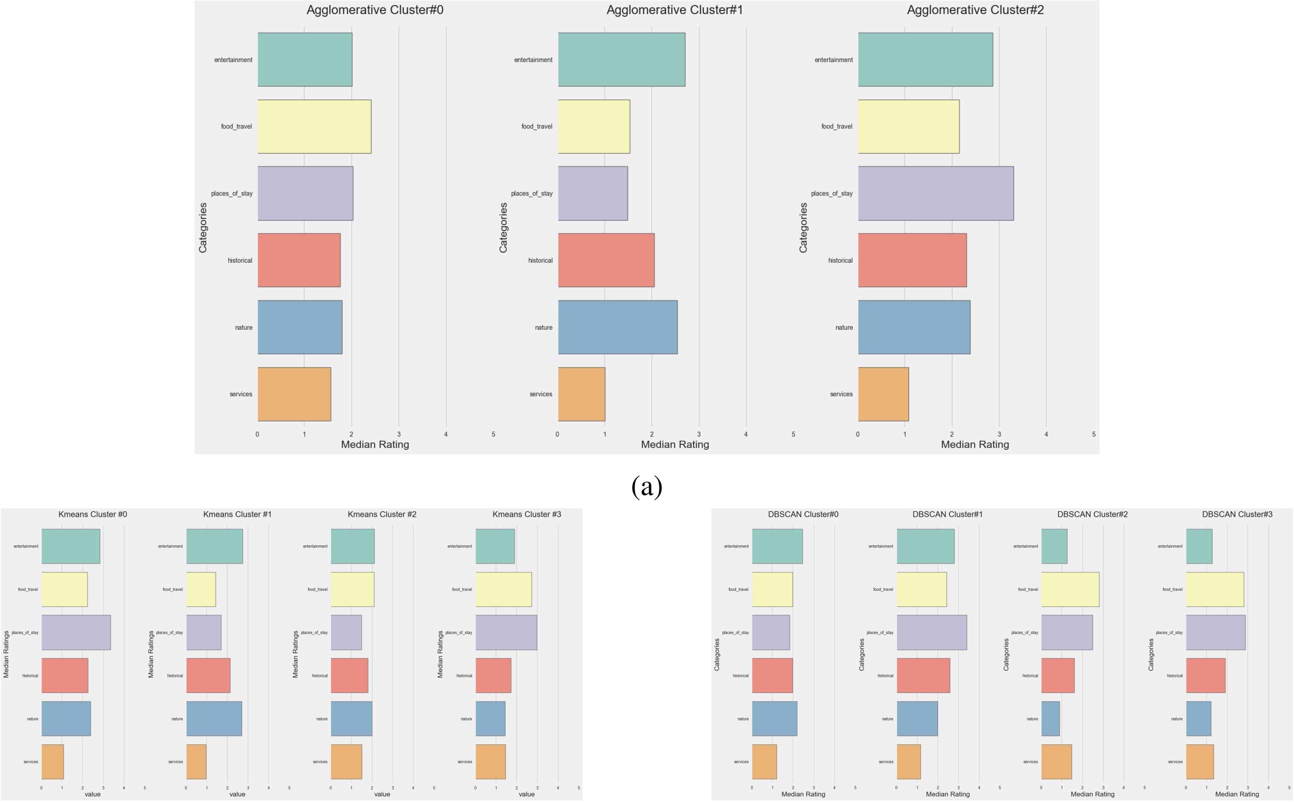
Moving on to K-means, the pattern is somewhat similar to Agglomerative, with the data again split into two primary groups based on satisfaction with the place of stay. Clusters 0 and 3 have higher ratings for accommodation, while Clusters 1 and 2 express dissatisfaction. However, the distinction between Cluster 0 and Cluster 3, as well as between Clusters 1 and 2, becomes more evident when examining their ratings for food, travel, and nature. These factors are particularly significant in differentiating the clusters within K-means, providing more nuanced sub-groupings based on individual preferences and experiences.

Finally, DBSCAN presents a more challenging interpretation due to the wide distribution of clusters. Cluster 0 captures the majority of the data and reflects individuals who are generally satisfied with most attractions, neither overly excited nor disappointed. However, the remaining clusters are much smaller and more specialized, representing niche preferences in areas such as satisfaction with the place of stay, food, travel, and nature. These smaller clusters suggest that DBSCAN identifies specific, perhaps more extreme, preferences that are not as widespread within the dataset.

As mentioned, a good way to assess how well our clustering algorithm performed after applying PCA is by examining three key metrics: the silhouette score, the Calinski-Harabasz index, and the Davies-Bouldin index, all of which were discussed in the previous section. Each algorithm's chosen number of clusters represents the best result for that specific method, but how do they compare against each other? To provide a clearer picture, the table in **Appendix A** summarizes the results, allowing us to evaluate the performance of each algorithm based on the metric we will use.

Looking at the Silhouette Score, K-means performs the best with a score of 0.391, indicating the most well-separated and cohesive clusters. Agglomerative follows with a score of 0.349, suggesting slightly less distinct clusters compared to K-means, with more overlap between them. DBSCAN, with a score of 0.04, shows poor cluster separation, likely due to its creation of one dominant cluster (Cluster 0) that captures most of the data, along with several smaller, less cohesive clusters. This result was expected given DBSCAN's tendency to form large, less distinct clusters, which was evident in the PCA plot, where the clusters appeared to be spread out and overlapping.

When examining the Calinski-Harabasz Index, K-means again outperforms the other algorithms with a score of 4404.151, reflecting good balance between cluster cohesion and separation. Agglomerative Clustering comes next with a score of 3785.521, which indicates reasonable separation but not



(a)

(b)

(c)

Figure 15: Median ratings for individual features across the clusters generated by the Agglomerative (a), K-Means (b), and DBSCAN (c) Clustering Algorithms.

as strong as K-means. DBSCAN's score of 123.27 highlights its less favorable clustering structure, with a large central cluster and smaller, possibly incoherent clusters that reduce overall separation. Again, this was anticipated based on the PCA plot, where DBSCAN's results showed a more chaotic distribution.

In the Davies-Bouldin Index, K-means continues to lead with a score of 0.818, showing that its clusters are more distinct and less similar to one another. Agglomerative follows with a slightly higher value of 0.919, indicating some overlap, though still relatively distinct clusters. DBSCAN scores the highest at 2.12, suggesting poor separation and a less meaningful cluster structure, primarily due to the dominance of the large central cluster and the unclear smaller clusters. This aligns with our expectations based on the PCA plot, where DBSCAN's cluster distribution appeared less defined.

In conclusion, K-means stands out as the best-performing algorithm across all three metrics, producing the most cohesive and well-separated clusters. Agglomerative Clustering is effective but falls short of K-means, while DBSCAN shows the weakest performance overall, with poorly defined clusters and less meaningful separation. These results emphasize the importance of selecting the appropriate clustering algorithm based on the data's characteristics and the analysis objectives. As anticipated, DBSCAN's performance was hindered by the overlapping clusters visible in the PCA plot, which made it less suitable for this particular dataset.

5 Discussion

5.1 Summary and Limitations

Overall, it seems that the clustering is improved a little by the addition of dimensionality reduction, with agglomerative generally having better silhouette score and K-means generally having better Calinski-Harabasz. DBSCAN had generally poor silhouette and Calinski-Harabasz but sometimes had passable Davies Bouldin index. However, most of these scores are not objectively high, indicating that either our data or algorithms were not sufficient to find very meaningful cluster sets (though some clusters do capture real trends).

Our main limitation was the lack of labels in the provided data, which greatly limited the scope of our research and our possibilities for evaluating model performance. Still, we may be able to improve our analysis by investigating even more methods. Potentially relevant dimensionality reduction techniques not covered include kernel PCA, locally linear embedding (a relative of Isomap), or autoencoders (neural networks). For clustering, our use case seems possibly conducive to affinity propagation, bisecting k-means (a combination of hierarchical and k-means), and HBDSCAN (an ensemble variant of DBSCAN).

5.2 Potential Applications

1. **Cross-Brand Opportunities:** Clustering insights can also create opportunities for collaboration between businesses with overlapping customer bases. For instance, two different companies where one specializing in museums and another in fine dining might find they serve the same cluster of culturally oriented travelers. They could offer bundled promotions or reciprocal discounts to encourage cross-patronage. These kinds of strategic partnerships are already seen in industries like retail and hospitality, and clustering methods offer a more data-driven approach to identifying the most promising collaborations.

2. **Tourism Planning and Policy Development:** Governments and tourism boards can leverage clustering insights to better understand the diverse interests of travelers and plan accordingly. If certain regions attract clusters that prioritize cultural experiences, infrastructure investments might focus on museums, historical sites, or festivals. In contrast, regions popular among adventure-seeking clusters could benefit more from developing outdoor recreation or eco-tourism options. By aligning public investment and marketing strategies with data-backed tourist preferences, cities and countries can optimize visitor satisfaction while boosting economic returns.

3. **Personalized Travel Recommendations:** Clustering can also enhance the capabilities of travel apps and online booking platforms. By identifying user segments based on attraction preferences, these platforms can offer personalized itineraries, attraction suggestions, or package deals that align with a traveler's interests. For example, someone falling into a cluster with high ratings for nightlife and shopping might receive recommendations for vibrant cities, evening events, and popular shopping districts. This kind of tailored experience can improve user satisfaction, increase engagement, and drive conversions for platforms and partner businesses alike.

References

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Kumar, S., & Toshniwal, D. (2016). Analysis of hourly road accident counts using hierarchical clustering and cophenetic correlation coefficient (cpcc). *J Big Data*, *3*(1), 13. https://doi.org/10.1186/s40537-016-0046-3

Renjith, S. (2018). Travel review ratings [Online. Available: https://doi.org/10.24432/C5C31Q].

Ahn, D., Park, H., & Yoo, B. (2020). Which group do you want to travel with? a study of rating differences among groups in online travel reviews. *Tourism Management*, *78*, 104046. https://doi.org/10.1016/j.tourman.2019.104046

Statistics Canada. (2025, March). Table 36-10-0232-01: Jobs generated by tourism (x 1,000) [Quarterly data. Formerly CANSIM 387-0003. Released March 27, 2025. Accessed April 9, 2025]. https://doi.org/10.25318/3610023201-eng

Appendix A

Grouping Only	**Agglomerative**	**K-Means**	**DBSCAN**
Number of Clusters	2	2	3
Silhouette Score	0.309	0.253	0.160
Calinski-Harabasz Score	14.812	1790.145	33.294
Davies Bouldin	0.837	1.617	0.796

Table 1: Clustering results for Grouping Only

PCA	**Agglomerative**	**K-Means**	**DBSCAN**
Number of Clusters	3	4	4
Silhouette Score	0.349	0.391	0.04
Calinski-Harabasz Score	3785.521	4404.151	123.27
Davies Bouldin	0.919	0.818	2.12

Table 2: Clustering results for PCA

MDS	**Agglomerative**	**K-Means**	**DBSCAN**
Number of Clusters	3	4	12
Silhouette Score	0.445	0.433	0.024
Calinski-Harabasz Score	3682.255	5603.573	593.630
Davies Bouldin	0.752	0.781	1.58

Table 3: Clustering results for MDS

5-Neighbor Isomap	**Agglomerative**	**K-Means**	**DBSCAN**
Number of Clusters	3	2	27
Silhouette Score	0.539	0.495	-0.097
Calinski-Harabasz Score	5840.046	7954.179	754.68
Davies Bouldin	0.610	0.713	0.623

Table 4: Clustering results for 5-Neighbor Isomap