1. Word count (excluding headers)

- 1062 words

2. Introduction

The objective of this report is to segment traveling customers using a quantitative approach based on a demographic dataset with 2000 observations across six variables: Biological Gender, Marital Status, Age, Education Level, Income Level, Occupation, and Settlement Size. Unsupervised machine learning techniques, specifically clustering, are used due to the absence of predefined target labels. The methodology includes Exploratory Data Analysis (EDA), standardization of numeric variables, and determining the optimal number of clusters using the Elbow method and Silhouette charts. Next, 2 clustering techniques-K-means++ (choosing centroids first and locating neighboring data) and Agglomerative clustering (treating each observation as a cluster and merging them)- are applied. Finally, the report suggests tailored marketing strategies for each segment.

3. Exploratory Data Analysis

Table 1: Variable description

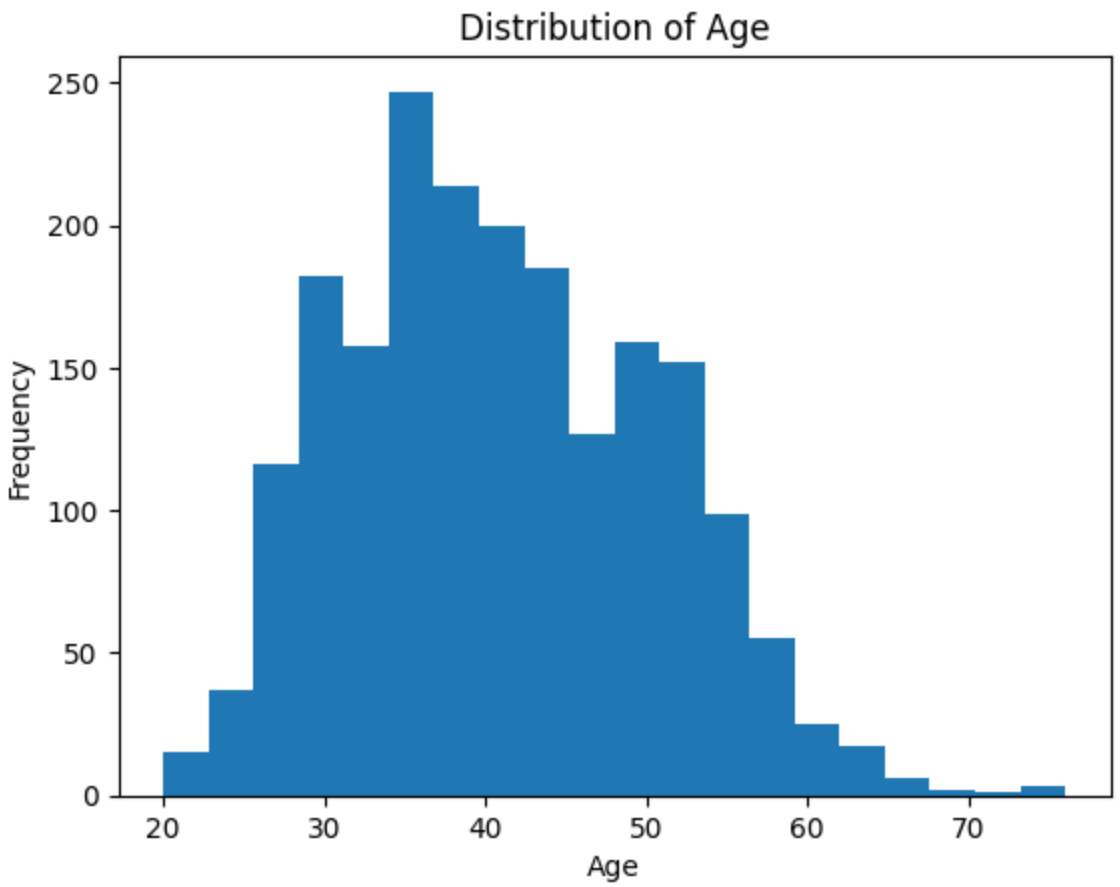
Variable name	Surveyed values	Data type
Biological Gender	Male (1) or Female (2)	Nominal
Marital Status	Single (0) or non-single (divorced / separated / married / widowed) (1)	Nominal
Age	Calculated as current year minus the year of birth of the customer at the time of creation of the dataset	Numeric
Education Level	Other/ Unknown (0), Highschool (1), University (2), or Graduate school (3)	Ordinal
Income Level	Self-reported annual income in US dollars of the customer	Numeric
Occupation	unemployed / unskilled (0), skilled employee / official (1), or management / self-employed / highly qualified employee / officer (2)	Ordinal
Settlement Size	small city (0), mid-sized city (1), big city (2)	Nominal

The general statistics of the dataset can be seen from the table below:

	Gender	Marital Status	Age	Education	Income	Occupation	Settlement Size
count	2000.00000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	0.60450	0.500500	40.823500	1.456500	137516.196500	0.612500	0.834000
std	0.48908	0.500125	9.455848	0.783846	46184.296588	0.674219	0.967942
min	0.00000	0.000000	20.000000	0.000000	35832.000000	0.000000	0.000000
25%	0.00000	0.000000	33.000000	1.000000	101262.750000	0.000000	0.000000
50%	1.00000	1.000000	40.000000	1.000000	133004.000000	1.000000	0.000000
75%	1.00000	1.000000	48.000000	2.000000	171232.500000	1.000000	2.000000
max	1.00000	1.000000	76.000000	3.000000	309364.000000	2.000000	2.000000
mode	1.00000	1.000000	33.000000	1.000000	74476.000000	0.000000	0.000000

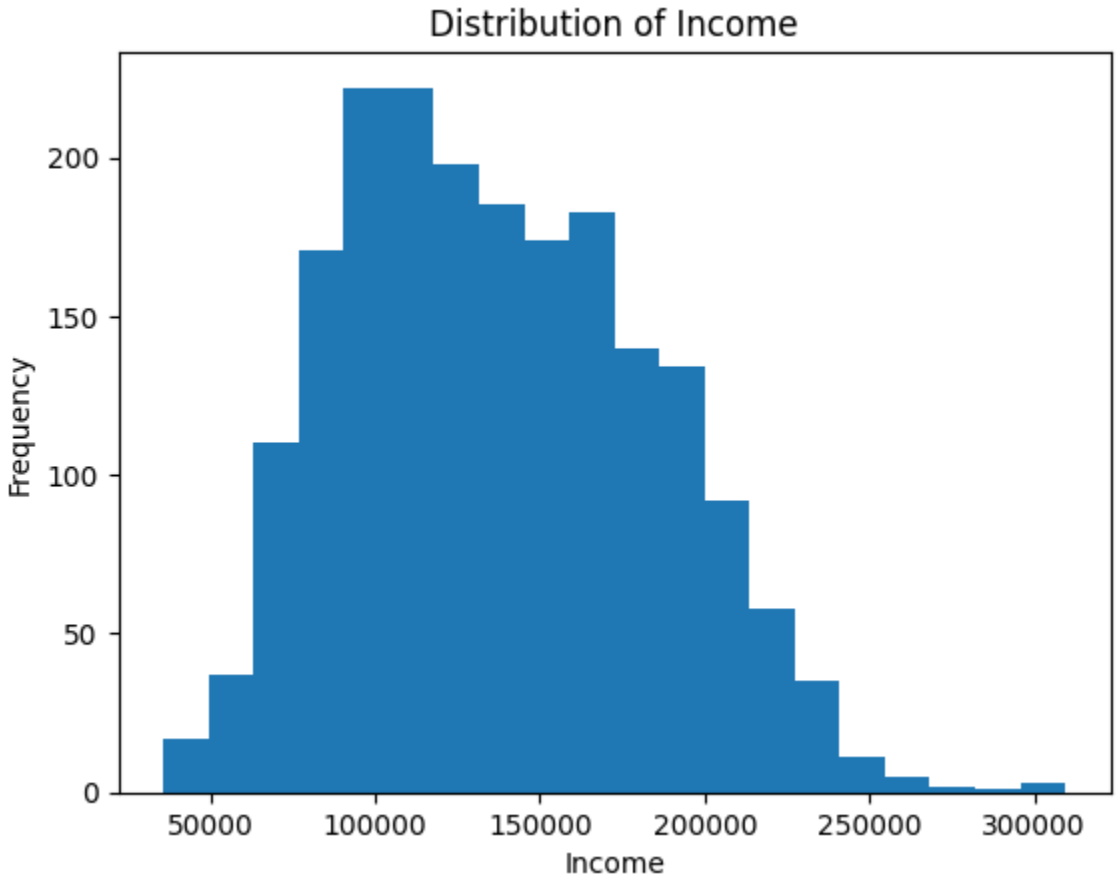
Preliminary statistics show a dominant number of respondents are male, non-single, 41 years old, having a high school graduate, earning an average of $137,516 USD annually, unemployed, and living in small cities. This indicates that major customers of this brand are from a lower to medium social background, implying that the data presenting is from a commodity product.

The age distribution of the customer is right-skewed (most frequent age of 33, mean age of) 41, suggesting that the brand attracts a relatively young demographic. Together with the commodity insight, this may mean that this product is more tailored to youngsters (maybe due to health issues or preferences)



Similarly, the income distribution is right-skewed, suggesting that the brand's products likely appeal to medium-income individuals, potentially positioning them as essential or commodity-based items that provide value across income levels. High-income customers, though less frequent, may also find value in the brand’s offerings, suggesting a broad appeal with potential for upselling

premium options.



Similarly, the income distribution is right-skewed, suggesting that most customers have medium-to-low incomes. This conclusion aligns with the notion that the brand's products are likely targeted at a broad audience, valuing affordability and convenience. Moreover, the negative relationship between income and demand reinforces the hypothesis that the service may be considered inferior, as its consumption seems to decrease as income levels rise.

4. Customer Segmentation

The first step for customer segmentation is standardization for numeric variables- Age and Income:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

# Standardize Age
np.set_printoptions(precision=3, suppress = True)
Age_scaled = sc.fit_transform(df_segment_scaled[['Age']])
df_segment_scaled['Age']=Age_scaled

#Standardize Income
Income_scaled = sc.fit_transform(df_segment_scaled[['Income']])
df_segment_scaled['Income']=Income_scaled
```

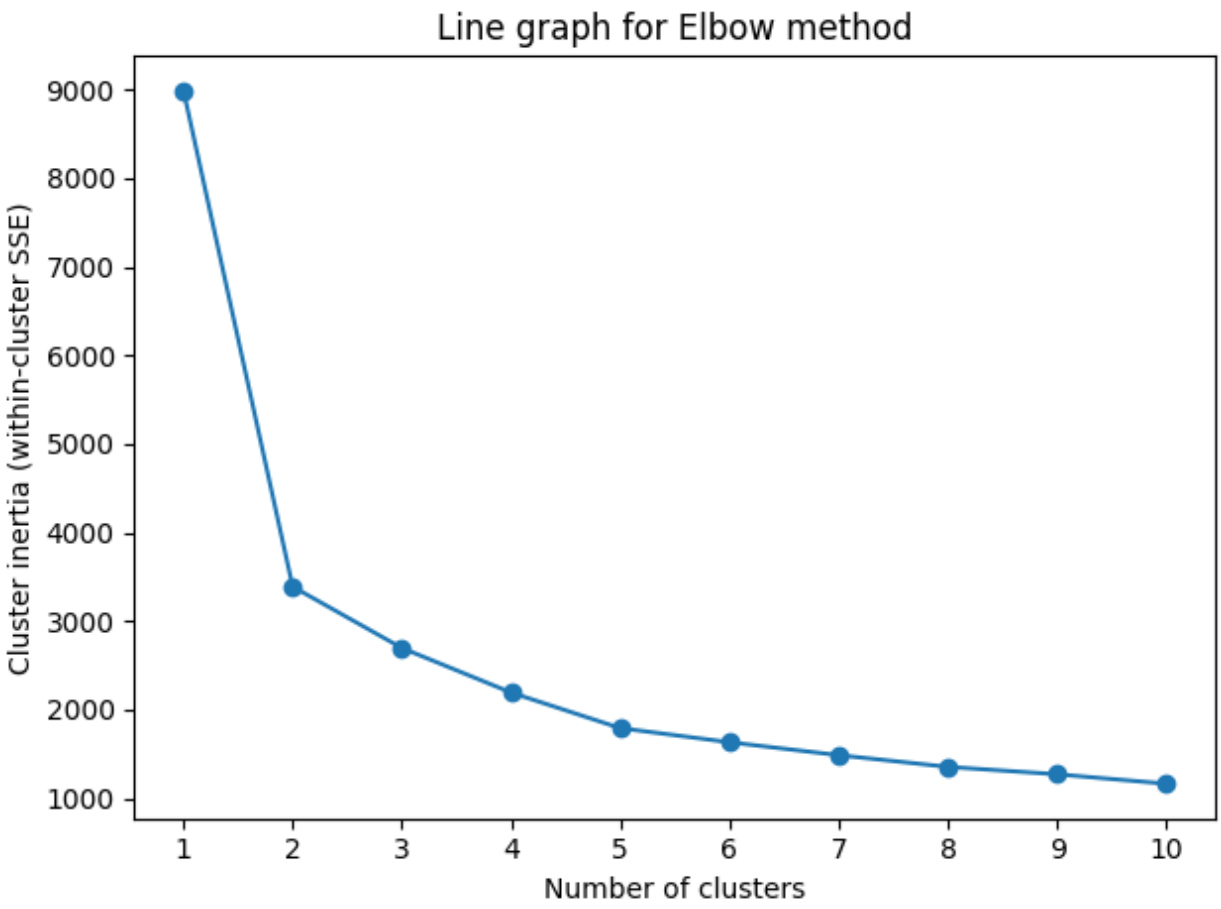
✓ 0.0s

Secondly, we will need to determine the optimal number of clusters using Elbow method and Silhouette Plots.

Firstly, Elbow methods determine the number of clusters (k) based on inertia attribute (sum of squared distances of samples to their closest cluster center). Specifically, k is chosen where inertia stops decreasing rapidly. The code is:

```
#finding optimal cluster number with Elbow method
from sklearn.cluster import KMeans
inertias = [] # empty list
for i in range(1, 11):
    km = KMeans(n_clusters=i,
                init='k-means++',
                n_init=10,
                max_iter=300,
                random_state=0)
    km.fit(df_segment_scaled)
    inertias.append(km.inertia_)

plt.plot(range(1, 11), inertias, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Cluster inertia (within-cluster SSE)')
plt.xticks(range(1,11))
plt.tight_layout()
plt.show()
#choose 2 as the optimal number of cluster centeroids
```

The output is:



Based on the model output, the report writer chooses 2 as the optimal number of clusters due to its highest slope, indicating the highest change in cluster inertia.

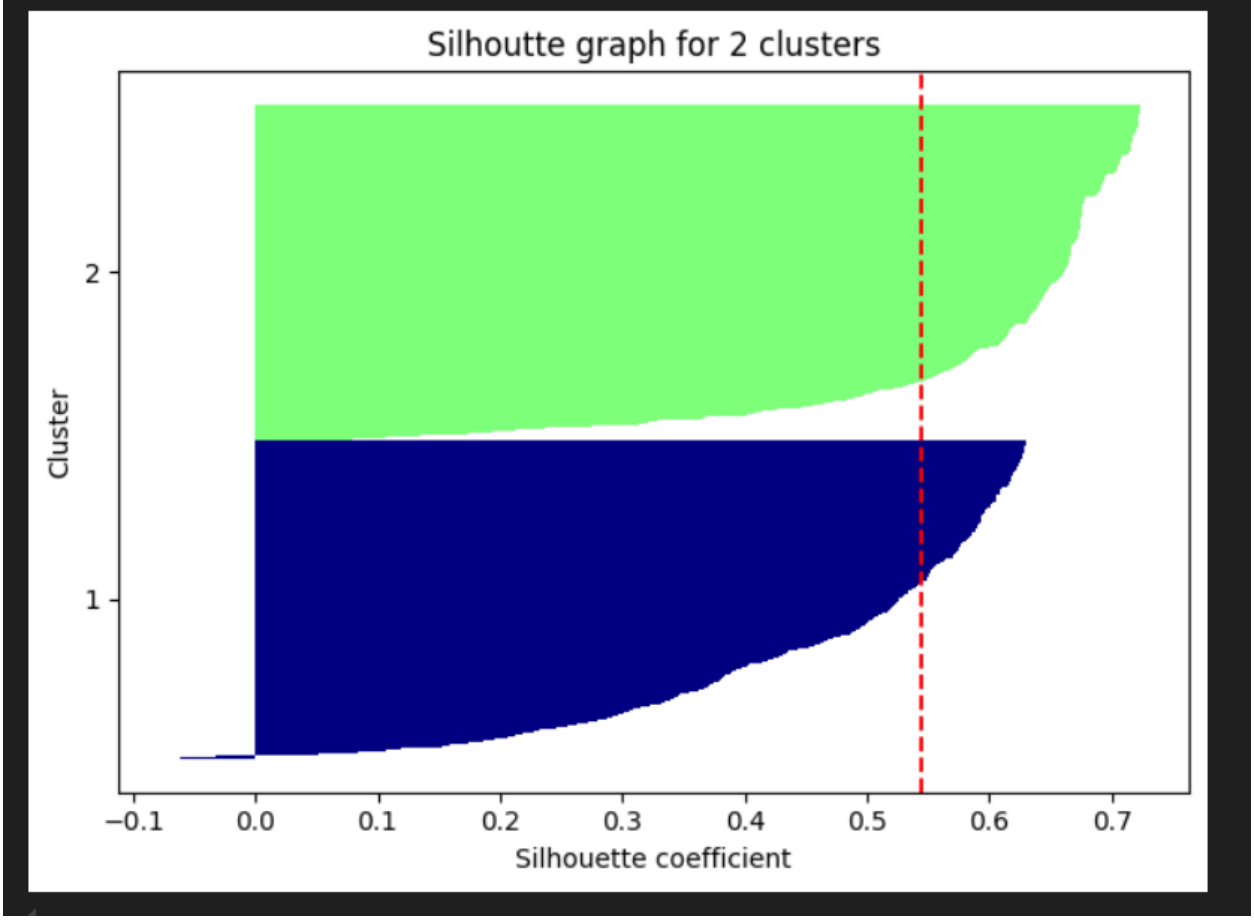
Silhouette charts are another way to determine optimal cluster number. The idea is to go through some numbers of clusters and choose the one with the highest Silhouette Coefficient. Under this scenario, we will create 3 Silhouette graphs with the number of clusters of 2,3 and 4. The code is:

```python
#finding optimal cluster number with silhoutte graphs
import numpy as np
from matplotlib import cm
from sklearn.metrics import silhouette_samples
for n_clusters in [2,3,4]:
    km = KMeans(n_clusters=n_clusters,
                init='k-means++',
                n_init=10,
                max_iter=300,
                tol=1e-04,
                random_state=0)

    y_km = km.fit_predict(df_segment_scaled)
    # print(y_km)
    cluster_labels = np.unique(y_km)
    # print(cluster_labels)
    n_clusters = cluster_labels.shape[0]
    silhouette_vals = silhouette_samples(df_segment_scaled, y_km, metric='euclidean')

## ------- plotting silhouette values -------
    y_ax_lower, y_ax_upper = 0, 0
    yticks = []
    for i, c in enumerate(cluster_labels):
        c_silhouette_vals = silhouette_vals[y_km == c]
        c_silhouette_vals.sort()
        y_ax_upper += len(c_silhouette_vals)
        color = cm.jet(float(i) / n_clusters)
        plt.barh(range(y_ax_lower, y_ax_upper), c_silhouette_vals, height=1.0, edgecolor='none', color=color)
        yticks.append((y_ax_lower + y_ax_upper) / 2.)
        y_ax_lower += len(c_silhouette_vals)
    silhouette_avg = np.mean(silhouette_vals)
    print(f'Average silhouette coefficient: {silhouette_avg:.2f}')
    plt.title(f"Silhoutte graph for {n_clusters} clusters")
    plt.axvline(silhouette_avg, color="red", linestyle="--") # plot vertical average line

    plt.yticks(yticks, cluster_labels + 1)
    plt.ylabel('Cluster')
    plt.xlabel('Silhouette coefficient')

    plt.tight_layout()
    plt.show()
```

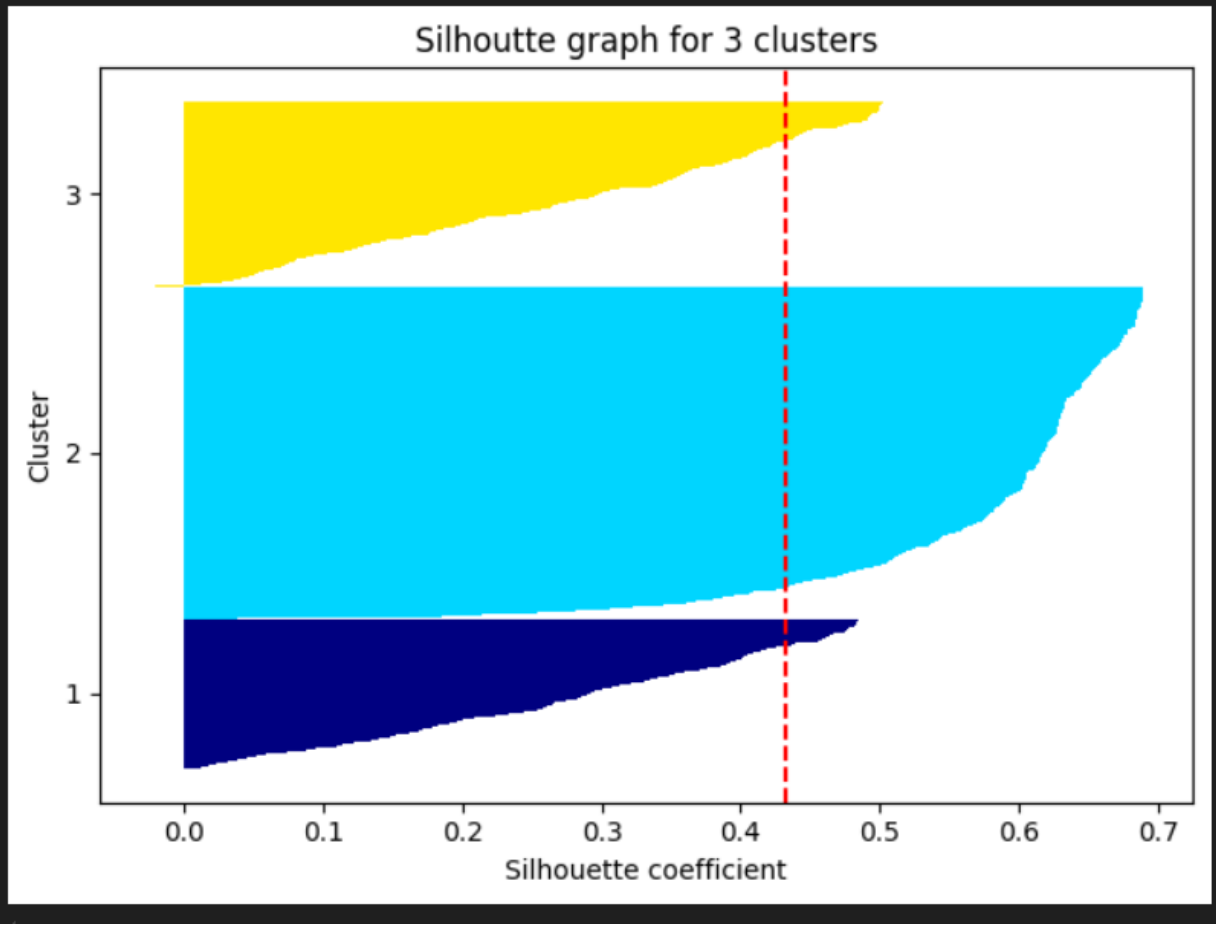
✓ 4.1s

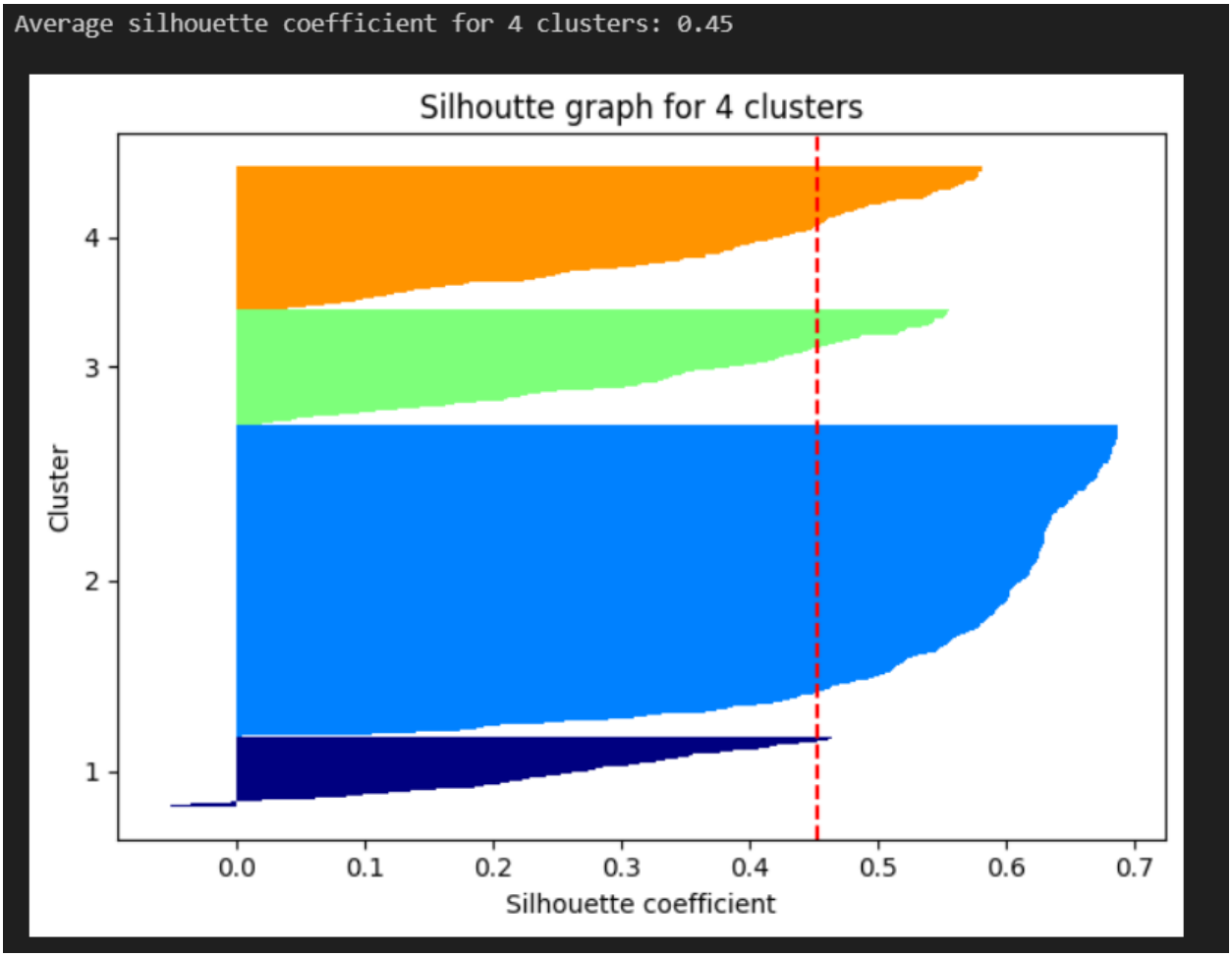
The outputs are:

```
Average silhouette coefficient for 2 clusters: 0.54
```



```
Average silhouette coefficient for 3 clusters: 0.43
```





Based on the charts, 2 conclusions are made. Firstly, all 3 clusters achieve Silhouette coefficients not close to 0, indicating that the dataset has been appropriately clustered. Secondly, coefficients are decreasing when k increases, meaning that the Silhouette coefficient is the largest at k=2. Thus, we can conclude that k=2 is the optimal number of clusters, matching with the decision of Elbow Method.

After choosing segmentation numbers, we will start to cluster. The process starts with choosing the initial cluster centroids, which can be done in 2 ways- K-means++ (top-down approach) and Agglomerative Clustering (bottom-up approach). Firstly, K-Means++ chooses initial centroids sequentially such that they have high probability of being far away from each other (not belong in the same cluster).

The code is:

```
#estimate clusters using K-Means++ technique
km_plus = KMeans(n_clusters=2,
                 init='k-means++',  # use starndard k-means rather than k-means++ (see below)
                 n_init=10,      # run 10 times with different random centroids to choose the final model with the lowest SSE
                 max_iter=300,   # max number of iterations for each run
                 random_state=0)
y_km_plus = km_plus.fit_predict(df_segment_scaled)
print('Cluster labels: %s' % y_km_plus)
```
✓ 0.1s

The output is:

```
Cluster labels: [0 1 1 ... 0 1 0]
```

On the other hand, Agglomerative clustering follows the steps of treating each data point as an individual cluster and iteratively merges the two closest clusters, updating the distance matrix, and repeating until all points are combined into one cluster.

The code is:

```
#estimate clusters using Agglomerative Clustering technique
from sklearn.cluster import AgglomerativeClustering

# ---------- 2 clusters --------------
ac = AgglomerativeClustering(n_clusters=2,
                             metric='euclidean',
                             linkage='complete')
labels = ac.fit_predict(df_segment_scaled)
print('Cluster labels: %s' % labels)
```
✓ 0.0s

The output is:

```
Cluster labels: [0 1 1 ... 0 1 0]
```

To get the cluster centroids, we will compute the representative data of each variable for each cluster 0 and 1. Specifically, Income and Age will be treated with average value while the other variables are treated with the most frequent data (mode). The tables below summarize the cluster centers and number of customers in each cluster:

For K-means++:

```
#Create a table for cluster centeroids of K means++ technique

df_segment['K Means classification']=y_km_plus.tolist()
df_segment_0=df_segment[df_segment['K Means classification']==0]
df_segment_1=df_segment[df_segment['K Means classification']==1]

numerical_data=['Age','Income']
KMP_cluster={}
for i in df_segment.columns:
        if i in numerical_data: KMP_cluster[i]=df_segment_0[i].mean()
        else: KMP_cluster[i]=df_segment_0[i].mode()[0].astype(int)
KMP_cluster_0=pd.DataFrame(KMP_cluster, index=[0])

KMP_cluster={}
for i in df_segment.columns:
        if i in numerical_data: KMP_cluster[i]=df_segment_1[i].mean()
        else: KMP_cluster[i]=df_segment_1[i].mode()[0].astype(int)
KMP_cluster_1=pd.DataFrame(KMP_cluster, index=[0])

KMP_cluster=pd.concat([KMP_cluster_0,KMP_cluster_1],axis=0)
KMP_cluster['Numer of customers']=[len(df_segment_0),len(df_segment_1)]

KMP_cluster
```

✓ 0.0s

Table 2: Cluster centroids for K-means++

	Gender	Marital Status	Age	Education	Income	Occupation	Settlement Size	K Means classification	Numer of customers
0	1	1	48.178279	2	173460.689549	1	2	0	976
0	0	0	33.813477	1	103256.601562	0	0	1	1024

For Agglomerative clustering:

```
#Create a table for cluster centeroids of Agglomerative Clustering technique

df_segment['K Means classification']=labels.tolist()
df_segment_0=df_segment[df_segment['K Means classification']==0]
df_segment_1=df_segment[df_segment['K Means classification']==1]

numerical_data=['Age','Income']
Agglo_cluster={}
for i in df_segment.columns:
        if i in numerical_data: Agglo_cluster[i]=df_segment_0[i].mean()
        else: Agglo_cluster[i]=df_segment_0[i].mode()[0].astype(int)
Agglo_cluster_0=pd.DataFrame(Agglo_cluster, index=[0])

Agglo_cluster={}
for i in df_segment.columns:
        if i in numerical_data: Agglo_cluster[i]=df_segment_1[i].mean()
        else: Agglo_cluster[i]=df_segment_1[i].mode()[0].astype(int)
Agglo_cluster_1=pd.DataFrame(Agglo_cluster, index=[0])

Agglo_cluster=pd.concat([Agglo_cluster_0,Agglo_cluster_1],axis=0)
Agglo_cluster['Numer of customers']=[len(df_segment_0),len(df_segment_1)]

Agglo_cluster

✓ 0.0s
```

Table 3: Cluster centroids for Agglomerative clustering

	Gender	Marital Status	Age	Education	Income	Occupation	Settlement Size	K Means classification	Numer of customers
0	1	1	47.080825	2	168085.081685	1	2	0	1163
0	0	0	32.129032	1	95041.150538	0	0	1	837

Based on 2 table outputs, the profile of each cluster for each technique is:

Table 4: Cluster centroids for K-means++

Feature	Cluster 0	Cluster 1
Gender	Male	Female
Marital Status	Non-single	Single
Age	Around 48	Around 34
Education	University	High School
Income	About 173460 USD/year	About 130257 USD/year
Occupation	Skilled employee/ office workers	Unemployed/ Unskilled
Settlement Size	Big cities	Small cities
Number of customers	976	1024

Table 5: Cluster centroids for Agglomerative clustering

Feature	Cluster 0	Cluster 1
Gender	Male	Female
Marital Status	Non-single	Single
Age	Around 47	Around 32
Education	University	High School
Income	About 168085 USD/year	About 95041 USD/year
Occupation	Skilled employee/ office workers	Unemployed/ Unskilled
Settlement Size	Big cities	Small cities
Number of customers	1163	837

We can see that there are a lot of overlaps between clustering using the 2 techniques, especially among non-numerical variables. Specifically, there are a total of 1813 overlap results between 2 techniques, indicating there are minor differences among the clustering results.

The overlap summary:

```
There are 976 overlaps in clustering segmentation 0
There are 837 overlaps in clustering segmentation 1
There are a total of 1813 overlaps
The number of differences between 2 techniques is 187
```

6. Recommendations

For segment 0, a premium positioning strategy is the most effective. Marketing should emphasize luxury feelings and family comfortability when traveling. Channels like LinkedIn and professional financial blogs are well-suited to this segment, given their high engagement with professional content. Besides, Exclusive offers, loyalty programs, and partnerships with luxury brands can reinforce the service's premium image. Moreover, Messaging should focus on responsibility, success, and wealth accumulation, aligning with their goals of family security and stability.

In contrast, segment 1 requires a more cost-centered marketing approach. The strategy should emphasize affordability and accessibility to travelling. Platforms such as Instagram and TikTok are ideal for engaging this group through relatable content and influencer partnerships. Promotions like bulk pricing, flexible payment plans and referral programs are highly recommended. Furthermore, messaging should center on empowerment, independence, and financial control, reflecting this segment's need for budget-friendly solutions. Collaborations with local organizations or community events will further build trust and connection, especially in smaller urban settings, making the product more approachable.

7. Conclusion

In conclusion, the segmentation analysis using K-means++ and Agglomerative Clustering has highlighted two distinct customer groups with similar market demand yet differing in needs and

preferences. By targeting the older, high-income segment through knowledge-driven channels and addressing their higher-level needs, while engaging the younger, average-income segment through social media and cost-focused messaging, businesses can effectively reach both groups. This tailored approach ensures that marketing strategies align with each segment's unique characteristics, maximizing engagement and driving long-term success.