Enterprise Data Science and Analytics

Data Science and Machine Learning

BookMe PROJECT REPORT

April 2022

Daniel Machado - 20211883

Diogo Torneiro - 20211770

Ricardo Carvalho - 20211821

Ricardo Nobre - 20211824

Vasco Pombo - m20211301

Index

1. Business Understanding

BookMe operates in the hospitality sector, providing accommodation and all necessary lodging services to leisure or business travelers through an international website. Although it possesses a sound customer network and stable revenue, efficient marketing campaigns and activities are needed to increase and maximize profit growth for the next three years.

The company aims to offer the best conditions possible, therefore at the end of each stay, customers rate their experience/satisfaction on a scale between 0 and 5, according to several indicators. The Data Science and Machine Learning team is going to use the collected data to elaborate clusters of clients, through an unsupervised learning technique that allows identifying critical aspects of the customers' features, while defining Client and Business perspectives, focused on the customer satisfaction and product usage respectively.

2. Data Understanding

Following the CRISP-DM methodology, the data set contains 15589 observations and 21 variables (fig.1 & fig.2), the latter can be classified in five categorical nominal, one numerical continuous and fifteen numerical discrete (fig.3).

2.1 Data Set Description

A preliminary analysis of the numerical variables denoted crucial insights pertinent to proceed in further analysis such as: "Year_Birth" which ranges between 1936 to 2014 and has missing values; "Checkout" has the highest average while "Comfort" the lowest; unlike all rating variables, "Wifi" has a maximum value of 6; "RewardPoints" has values in a different scale than the other variables (fig.4).

Additionally, for the categorical variables: an even split between clients who have churned or not can be seen(fig.5), there are at least 1000 clients with repeated names (fig.6), there is misclassification in "Longevity" (fig.7), more than half of customers travel in Business (fig.9) and few experienced the "RoomType" "Suite" (fig.10).

2.2 Skewness

Some variables in the data set are moderately negative skewed, i.e left tail distribution with mean<median<mode, these are: "Amenities", "Staff", "OnlineBooking", "PriceQuality", "RoomSpace", "CheckOut" and "Cleanliness"; the remaining can be considered approximately symmetric (fig.11). Consequently, as these numerical variables constitute evaluations from clients, here the skewness indicates that in these variables most evaluations are higher than the half of the available scale.

2.3 Kurtosis

Kurtosis demonstrates how heavily the tails of a distribution differ from the normal distribution, which can be particularly useful to indicate the small/large incidence of outliers. Unlike “RewardPoints”, all the variables have negative excess revealing flat tails and small incidence of outliers (fig.12). Nevertheless, the variable “RewardPoints” follows a platykurtic distribution with a positive excess kurtosis value of roughly 0.26, indicating heavy tails and a large incidence of outliers.

3. Data Preparation

After the understanding phase, the data needed to be prepared for clustering, hence multiple steps were followed.

3.1 Handling Missing Values

Even though there are several ways to fill missing values, such as using the mode or median for categorical and continuous variables respectively, the KNN imputer was the algorithm selected. From prior analysis, there were in total 195 missing values, solely in the variable ‘Year_Birth’ (Fig.13).

The mentioned algorithm includes the argument “n_neighbours”, which identifies the number of data points taken into consideration for the labeling of the missing value. In this case, since the mean (1981.71), mode (1982) and median (1981) are quite similar, it is possible to assume a somewhat normal distribution on this particular variable, therefore 2 was considered accurate to the argument, while it was assumed that error is stable and convergence reached. After executing the algorithm, the variable ended up with no missing values.

3.2 Duplicate Values

While checking the DataFrame for duplicate entries, 3 were found, these were considered to be wrong entries and removed from the Data Set.

3.3 Handling Outliers

There are several methods for handling outliers, the chosen one was IQR (Fig.14) with the assistance of Exploratory Data Analysis and Kurtosis. Firstly, a major variable candidate was identified, “RewardPoints” (fig. 15), which resulted in the removal by the IQR method of 293 observations.

Lastly, according to the Box Plots, “PriceQuality” and “CheckIn” were also candidates (Fig.16-17). However, for both variables too many observations would be removed, violating the rule of deleting more than 3% of the data, besides that the scale of the variables would be shortened and through Kurtosis analysis (fig.12) the number of outliers were not relevant. Consequently, several issues emerged, leading to the non removal of any observation in these variables.

3.4 Straight Lining Identification and Removal

Straight Lining occurs when a customer fills in the survey in a straight line, meaning equal answers in all categories. In this regard, 2 entries were found and removed since they were assumed to be filled in a rush or not truthfully (Fig.18).

3.5 Misclassifications and Reclassifications

In the exploration phase, two major misclassifications were identified that demanded reclassification: “y” values from “Longevity” were transformed to “Yes” (fig.8); “Wifi” had as maximum value 6 (fig.19), all those values were replaced by “5”, i.e. the maximum value of the ratings scale (fig.20).

3.6 Feature Engineering

The data set also allowed more features to be created: firstly, all observations started either with “Mr.” or “Mrs.”, hence “Gender” was created with values 0 for Females (Mrs.) and 1 for Males (Mr.), while “Name” was dropped from the data set; secondly, “Age” could be obtained subtracting the current year (2022) by “Year_Birth”, enabling “Year_Birth “ to be dropped as well; lastly, the mean of all rating variables in the data set, i.e. "OverallRating".

3.7 Dummy and Binary variables

Moreover, before modeling the remaining categorical variables needed to be transformed in dummies/binaries: firstly, in “Churn” 0 was assigned to “nochurn” and 1 to “churn”; secondly, in “Longevity” “no” was transformed to 0 and “yes” to 1; thirdly, in “TypeTravel” 0 was assigned to “leisure” and 1 to “business”; lastly, three options were available in “RoomType”, thus this resulted in the creation of two dummy variables “RoomType_single” and “RoomType_suite”, while the previous categorical variable was removed from the dataset.

3.8 Data Normalization

Since the algorithms used for the clustering phase work based on distances between data points, the data had to be normalized first to ensure no variable changed the results significantly.

Consequently, two different methods were considered for this problem. The first was StandardScaler, however that is more suited when most of the variables from the data set have normal distributions. Therefore, it was decided to go instead with MinMaxScaler, which scaled each variable between 0 and 1, by doing this the “RewardPoints” issue was solved.

3.9 Correlation Check

Last but not least, the correlation between variables was checked. Unquestionably, several variables have positive correlation, namely some have moderated positive correlation with range values between

0.6 and 0.8, including Comfort:FoodDrink, Comfort:BarService, Wifi:OnlineBooking, Wifi:BarService, Staff:OnlineBooking, Staff:BarService, OnlineBooking: BarService, PriceQuality:Cleanliness, Checkout:Cleanliness (fig.21).

Obviously, almost all of the rating variables have some significant positive correlation with "OverallRating", which is understandable because this feature is the mean of all others. Additionally, another correlation that is not moderated but is notable is between "RoomType_Single" and "TypeTravel".

4. Modeling

To get a better understanding of the results, the data was separated in two groups: "client_variables" and "business_variables".

For the "**client_variables**" group, the variables that have more similarity with client behavior were selected which were: "Longevity", "TypeTravel", "RewardPoints", "RoomType_single", "RoomType_suite", "Gender" and "Age".

For the "**business_variables**" group, the variables that are more related with the business/product itself were selected which were: "Churn", "Comfort", "ReceptionSchedule", "FoodDrink", "Location", "Wifi", "Amenities", "Staff", "OnlineBooking", "PriceQuality", "RoomSpace", "CheckOut", "CheckIn", "Cleanliness" and "BarService".

Besides the two above, a third group was created just for testing purposes called "**short_business_variables**" which contains the following variables: 'Churn' and 'OverallRating'. This alternative group was created to check how it would impact the final results versus using the rating variables.'

4.1 KMeans

After experimenting with several different clustering algorithms (mentioned further down), KMeans ended up being the one offering the best results. The algorithm was ran 4 times, first for the client variables subset of the dataframe, second for the business variables subset of the dataframe, third for the alternative variable group with 'Churn' and 'OverallRating' only, and finally, it was ran again to merge the results of the different perspectives.

Thus, in each perspective, the elbow method was applied to get the optimal number of clusters, while the upper and lower limits of the optimal number were tested.

4.1.1 Kmeans - Client perspective ("client_variables")

After applying the algorithm and getting the optimal number of clusters using the elbow method, the result was 5 (Fig 22) with the following distribution:

Cluster 0 - 2341, Cluster 1 - 3092, Cluster 2 - 3604, Cluster 3 - 3282, Cluster 4 - 2975 (details in Fig.23)

4.1.2 Kmeans - Business perspective ("business_variables")

After applying the algorithm and getting the optimal number of clusters using the elbow method, the result was 4 (Fig 24) with the following distribution:

Cluster 0 - 3277, Cluster 1 - 4194, Cluster 2 - 3028, Cluster 3 - 4795 (details in Fig.25-28)

3 and 5 clusters were also tested for this perspective but the results were not as great.

4.1.3 Kmeans - Short-Business perspective ("business_variables")

After applying the algorithm and getting the optimal number of clusters using the elbow method, the result was 2 (Fig 29), however, after testing with 3 and 4 clusters, and seeing better results, the number of clusters chosen ended up being 3 with the following distribution:

Cluster 0 - 7002, Cluster 1 - 4766, Cluster 2 - 3526 (details in Fig.30)

4.2. Other Clustering Techniques

4.2.1 Hierarchical Clustering

The Hierarchical Clustering was applied using an agglomerative method, by doing this it was evaluated the most appropriate linkage while keeping the Euclidean Distance constant. According to the dendrograms (fig. 31-41), in any of the perspectives approached the Ward linkage seems not only to be the case that provides the most reasonable number clusters, but also the one that goes more in accordance with the numbers of clusters obtained in K-Means.

Nevertheless, the characterization of the clusters (fig. 23, 25-28, 42-46) and the customers' distribution per cluster (fig 48-53) demonstrate differences for both Client and Business perspectives, it seems that in the Hierarchical method customers tend to focus more in a cluster rather than in K-Means. Obviously, this can disregard further analysis because when two different algorithms are executed containing the same data different results are obtained, so this is not relevant to report.

Consequently, there are in fact no significant changes in terms of both clustering techniques. However, the team is evaluating a big data problem, therefore K-Means is the appropriate model to proceed, due to the Hierarchical's limitations of being computationally intensive and the inability to go back after every iteration.

4.2.2 Gaussian Mixture Model (GMM)

The GGM was applied in an attempt to provide a more probabilistic and flexible approach in terms of the cluster assigned to each customer in the model. Considering the BIC criteria (fig. 54&58), it seems

that the number of clusters are in accordance with the K-Means technique for the Client and Short-Business perspective, which is in fact confirmed by the respective sound silhouette coefficients (fig. 55&59).

Nonetheless, the same rationality does not apply for the Business perspective, the BIC score is not lower enough (fig.56) and this is in fact confirmed by a weak silhouette score (fig.57). Consequently, the distance between clusters is not significant and an increase in the number of clusters to more than 4 would be necessary.

Thus, using K-Means or GMM would cause significant differences in the clusters of the model, but since the previously normalization method chosen was MinMax instead of Standard Scaler, the GMM should not be considered as the best alternative for the model and K-Means is once again maintained.

4.2.3 DBSCAN

Unlike K-Means, DBSCAN uses density to form clusters and it is not sensitive to outliers. Considering its arguments, several “min_sample” values were tested, while the eps was defined through k-NN to compute the average distance between each point and its k-nearest neighbors, with the k being the “min_sample” parameter. Right after, the distances were plotted in ascending order on a k-distance graph fig.60, where the eps would consist in the point with the highest slope, then noise points, number of clusters and respective number of clients were obtained. Although several combinations of features and parameters were used, unfortunately the results were always too many or too few significant clusters (0 or 1). When there were too many, the min_samples increased and it resulted in fewer clusters, but as expected it also resulted in most clients being treated as outliers (fig.60-62). Thus, density to cluster might not be a good fit due to the dataset nature.

4.3 Dimensionality Reduction with PCA

A dimensionality reduction technique that would reduce the number of rating variables would add value to the model, therefore the Principal Component Analysis was applied. Here, Pearson's rule was used to find an optimal number of PC’s. The results were 7 components, which explain 80% of the variance in the dataset (fig. 64 & 65).

Although good correlations between PC’s and variables were obtained, having 7 PC’s, where each PC can be more correlated with more than one rating variable, would make the final analysis of the dataset harder, so it was decided not to use them.

4.4 Merging Clusters and Conclusions (Fig.67 & 68)

After the application of the KMeans algorithm to the different sets of variables, the resulting clusters had to be merged in order for the team to create the desired marketing plan.

The elbow method was applied to this analysis as well (Fig.66), giving the team an optimal number of clusters of 4, however, after testing with 3 and 5 clusters, the results of 5 clusters were better to extract a more detailed customer segmentation plan.

4.4.1 Description of clusters:

Cluster 0 (size 2856)	**Cluster 1 (size 2341)**	**Cluster 2 (size 3518)**	**Cluster 3 (size 4782)**	**Cluster 4 (size 1797)**
Clients registered more than 1 year Clients with no churn Mainly women Travel in leisure High ratings Double room No Suite	Clients with churn Clients registered more than 1 year Mainly men Travel Leisure Double room No Suite Low average of ratings Value Reception Schedule more than others	New clients (registered less than a year) Churn clients Business travelers Mainly young women Double room No Suite Close to the lower bound of overall ratings	Best evaluation ratings Clients with no churn Clients registered more than 1 year Business travelers Heterogeneous gender and age Single room No suite	Churn clients Clients registered more than 1 year Business travelers Low ratings Mainly men Single room No suite

5. Evaluation and Marketing plan

Two strategies were planned for the marketing plan, one for the current customers and how their experience could be improved when visiting any hotel, and another to attract new customers who might be the ideal client for the business.

5.1 Existing Client Strategy (use our own platforms to communicate with them)

Cluster 3 contains what would be called, the ideal customers, registered for more than one year and who give high ratings. Which can be assumed to be the customers who enjoy the business the most. These are mainly clients on business travel who stay in single rooms and have a high age average compared to the rest of the clients. These clients can be offered room upgrades from time to time to keep the relationship between business and customer always on the best terms.

Cluster 2 contains mostly female clients traveling on business, but registered for less than one year and who don't enjoy the service as much as the clients in cluster 3. These clients have the potential to become the ideal customers and can be offered discounts to stay in single rooms instead of doubles (which most of them do). These clients might value a room for themselves more and therefore give the whole service a better rating in the end.

Based on these 2 clusters, and to nurture future client relationships, campaigns can be created for customers/companies who have been registered for less than one year to receive discounts on booking single rooms for their employees.

Cluster 4 seems to contain clients who have been ideal clients but that don't enjoy the business anymore. These clients are important for the business because they can help BookMe understand what factors can influence long date customers to enjoy the business less. Discounts or room upgrades can be offered to these customers in exchange for more detailed input on what BookMe could improve on.

Regarding customers who use the service mostly for leisure travel (cluster 0 & 1), it can be noticed that the higher ratings come from cluster 0, and the higher rate of churn comes from clients in cluster 1. Future campaigns can be created to target women, because they seem to enjoy the business the most. However, BookMe should also contact their previous male customers who traveled for leisure to try and understand the gap in overall ratings scores between genders. To incentivize answers, discounts on future travels can be offered.

As a side note, 3 ratings have low overall scores independent of the type of client. These are 'Comfort', 'ReceptionSchedule' and 'FoodDrink'. Steps might be taken into consideration to boost these scores.

5.2 New Customers strategy (Use media platforms to find more lookalike customers)

Insights on campaigns to acquire future clients can also be formulated from the analysis.

Clients can be divided into 2 big groups: business and leisure. For business clients/companies, BookMe can target both genders, focusing on older people (around 40 years old), and offer discounts on single rooms, while for leisure clients, BookMe can target younger women and offer discounts on doubles or suites.

These future clients have the lowest likelihood of churn and can become good clients for the business.

6. Annexes

Variable	Description
Name	Customer's name
Year Birth	Customer's birth year
Longevity	Whether the customer registered more than 1 year ago or not
Churn	Whether the customer churned or not (churn or nochurn)
TypeTravel	Customer's reason for travelling (business or leisure)
RoomType	Type of room reserved
RewardPoints	Customer's rewarding point for loyalty
Comfort	Satisfaction level of customer regarding comfort of the room (0 to 5)
ReceptionSchedule	Satisfaction level of customer regarding reception schedule (0 to 5)
ReceptionSchedule	Satisfaction level of customer regarding food and drink available (0 to 5)
Location	Satisfaction level of customer regarding accommodation location (0 to 5)
Wifi	Satisfaction level of customer regarding wi-fi service (0 to 5)
Amenities	Satisfaction level of customer regarding accommodation amenities(0 to 5)
Staff	Satisfaction level of customer regarding staff (0 to 5)
OnlineBooking	Satisfaction level of customer regarding online booking ease(0 to 5)
PriceQuality	Satisfaction level of customer regarding price quality relationship (0 to 5)
RoomSpace	Satisfaction level of customer regarding room space (0 to 5)
CheckOut	Satisfaction level of customer regarding check-out (0 to 5)
CheckIn	Satisfaction level of customer regarding check-in (0 to 5)
Cleanliness	Satisfaction level of customer regarding cleanliness (0 to 5)
BarService	Satisfaction level of customer regarding bar service (0 to 5)

Figure 1 - Variables Description

```
df_original_row_size = df.shape[0]
df_original_columns_size = df.shape[1]

print("Dataframe has", df_original_row_size, "rows and", df_original_columns_size, "columns")
```

```
Dataframe has 15589 rows and 21 columns
```

Figure 2 - Dataframe size

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15589 entries, 0 to 15588
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Churn              15589 non-null  object
 1   Name               15589 non-null  object
 2   Longevity          15589 non-null  object
 3   Year_Birth         15394 non-null  float64
 4   TypeTravel         15589 non-null  object
 5   RoomType           15589 non-null  object
 6   RewardPoints       15589 non-null  int64
 7   Comfort            15589 non-null  int64
 8   ReceptionSchedule  15589 non-null  int64
 9   FoodDrink          15589 non-null  int64
 10  Location           15589 non-null  int64
 11  Wifi               15589 non-null  int64
 12  Amenities          15589 non-null  int64
 13  Staff              15589 non-null  int64
 14  OnlineBooking      15589 non-null  int64
 15  PriceQuality       15589 non-null  int64
 16  RoomSpace          15589 non-null  int64
 17  CheckOut           15589 non-null  int64
 18  Checkin            15589 non-null  int64
 19  Cleanliness        15589 non-null  int64
 20  BarService         15589 non-null  int64
dtypes: float64(1), int64(15), object(5)
memory usage: 2.5+ MB
```

Figure 3 - Type of variables in the Data Set

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Year_Birth	15394.0	1981.706444	15.179042	1936.0	1970.0	1981.0	1994.0	2014.0
RewardPoints	15589.0	5022.593816	1027.962379	409.0	4445.0	5088.0	5649.0	6950.0
Comfort	15589.0	2.841619	1.388624	0.0	2.0	3.0	4.0	5.0
ReceptionSchedule	15589.0	2.997242	1.518994	0.0	2.0	3.0	4.0	5.0
FoodDrink	15589.0	2.844570	1.436948	0.0	2.0	3.0	4.0	5.0
Location	15589.0	2.986016	1.299438	1.0	2.0	3.0	4.0	5.0
Wifi	15589.0	3.245109	1.327026	0.0	2.0	3.0	4.0	6.0
Amenities	15589.0	3.374816	1.352417	0.0	2.0	4.0	4.0	5.0
Staff	15589.0	3.506383	1.319565	1.0	3.0	4.0	5.0	5.0
OnlineBooking	15589.0	3.454231	1.310343	0.0	2.0	4.0	5.0	5.0
PriceQuality	15589.0	3.459683	1.268130	1.0	3.0	4.0	4.0	5.0
RoomSpace	15589.0	3.470845	1.293873	0.0	2.0	4.0	5.0	5.0
CheckOut	15589.0	3.700558	1.158644	1.0	3.0	4.0	5.0	5.0
Checkin	15589.0	3.327282	1.266872	1.0	3.0	3.0	4.0	5.0
Cleanliness	15589.0	3.692347	1.154437	1.0	3.0	4.0	5.0	5.0
BarService	15589.0	3.347360	1.300452	0.0	2.0	3.0	4.0	5.0

Figure 4 - Statistical Description of Numerical Variables

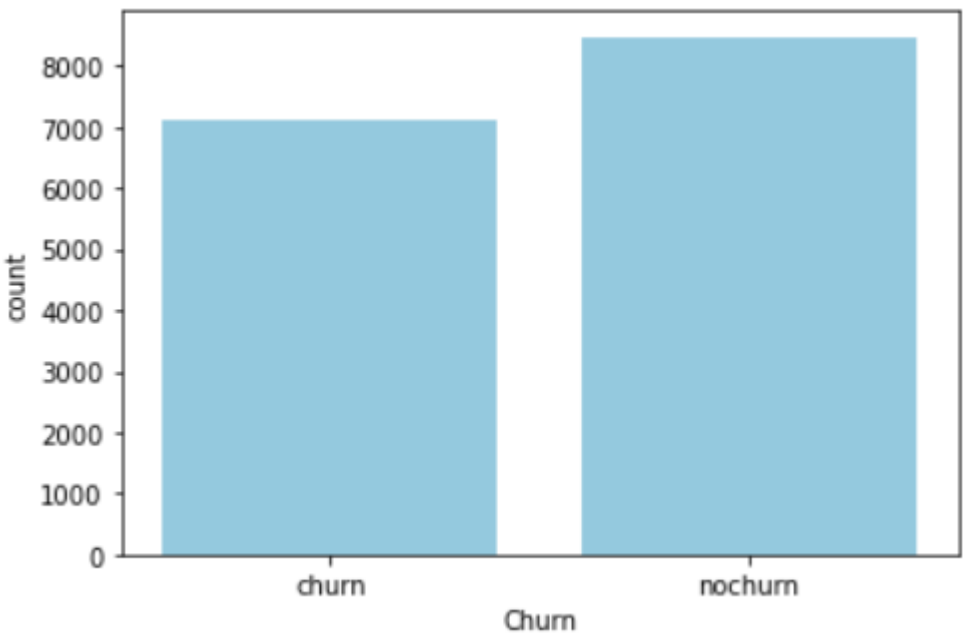

Figure 5 - Countplot for the categorical variable "Churn"

```
df['Name'].value_counts().nlargest(1000)
```

```
Mr. Michael Smith      9
Ms. Amanda Smith       7
Mr. John Smith         7
Mr. Michael Jones      6
Mr. William Smith      6
                      ..
Ms. Elizabeth Adams    2
Mr. Daniel Hall        2
Mr. William Wright     2
Ms. Sandra Lopez       2
Mr. Joseph Martinez    2
Name: Name, Length: 1000, dtype: int64
```

Figure 6 - Categorical Variable "Name" values

```
df['Longevity'].value_counts()
```

```
yes    12548
no      2874
y        167
Name: Longevity, dtype: int64
```

Figure 7 - Categorical Variable "Longetivity" values

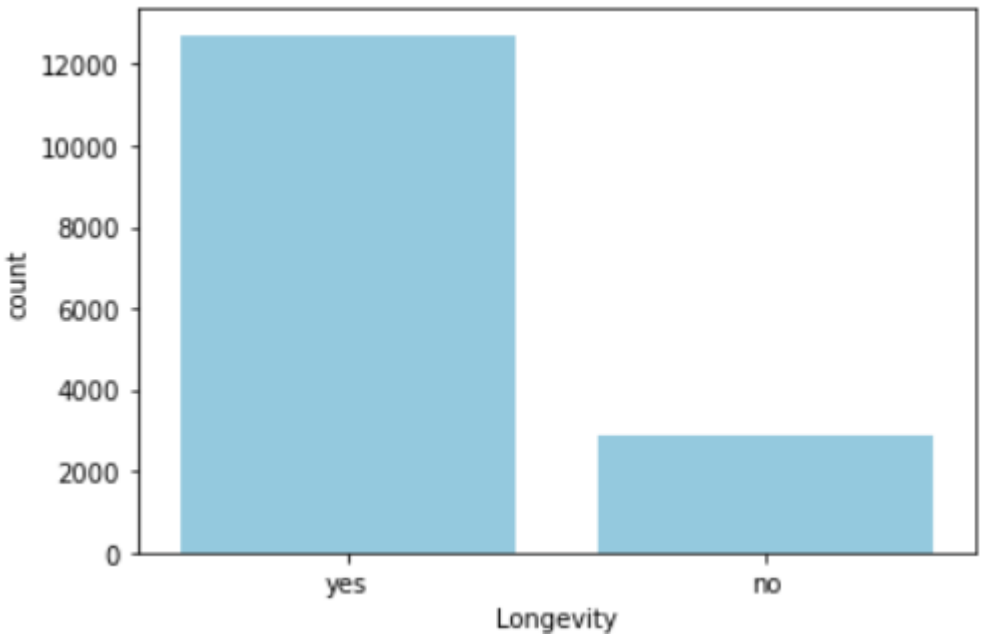

Figure 8 - Countplot for Categorical Transformed Variable "Longetivity"

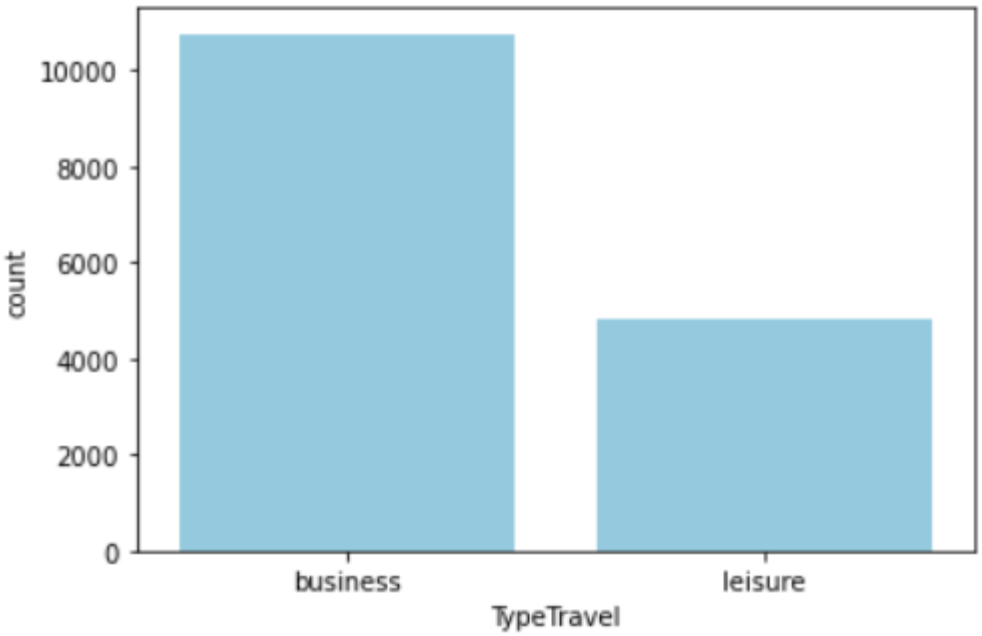

Figure 9 - Countplot for Categorical Variable "TypeTravel"

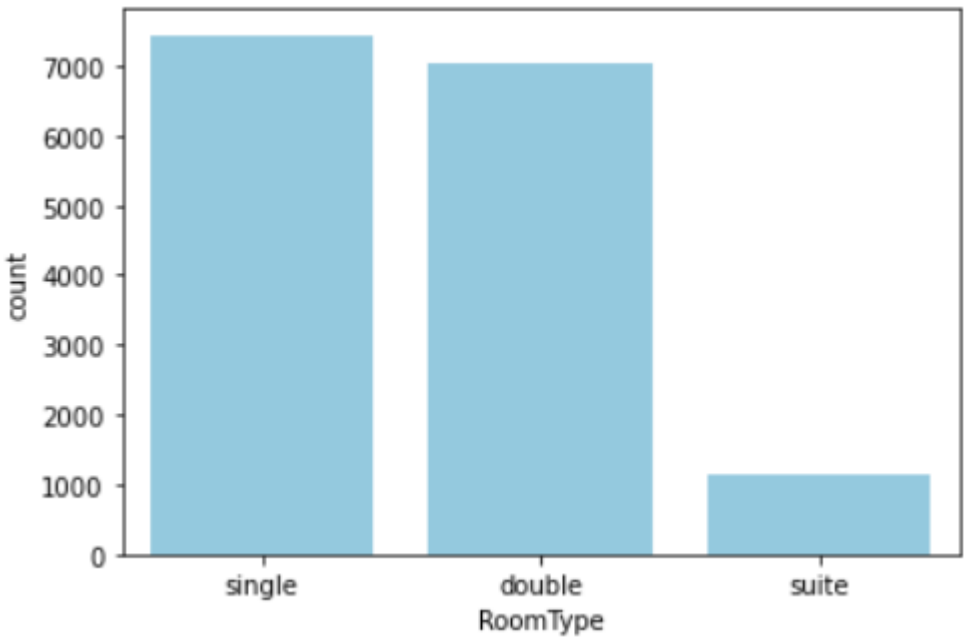

Figure 10 - Countplot for Categorical Variable "Room Type"

Year_Birth	-0.003847
RewardPoints	-0.453779
Comfort	-0.100907
ReceptionSchedule	-0.260705
FoodDrink	-0.123610
Location	-0.050229
Wifi	-0.183010
Amenities	-0.599498
Staff	-0.554561
OnlineBooking	-0.472074
PriceQuality	-0.503381
RoomSpace	-0.482952
CheckOut	-0.750689
Checkin	-0.382588
Cleanliness	-0.745131
BarService	-0.358297
Gender	0.048900

Figure 11 - Skewness of Numerical Variables

Wifi	-1.111251
ReceptionSchedule	-1.077243
Location	-1.076583
FoodDrink	-0.967047
BarService	-0.949765
Comfort	-0.935544
OnlineBooking	-0.931811
RoomSpace	-0.864644
Staff	-0.858889
Checkin	-0.812149
PriceQuality	-0.772663
Year_Birth	-0.729800
Amenities	-0.540657
Cleanliness	-0.225183
CheckOut	-0.225016
RewardPoints	0.260135

Figure 12 - Kurtosis of Numerical Variables

```
Churn                0
Longevity            0
Year_Birth         195
TypeTravel           0
RoomType             0
RewardPoints         0
Comfort              0
ReceptionSchedule    0
FoodDrink            0
Location             0
Wifi                 0
Amenities            0
Staff                0
OnlineBooking        0
PriceQuality         0
RoomSpace            0
CheckOut             0
Checkin              0
Cleanliness          0
BarService           0
Gender               0
```

Figure 13 - Variables with null values in the Data Set

```python
#method to return the boundaries of IQR
def get_IQR_bounds(s):
    q1 = s.quantile(0.25)
    q3 = s.quantile(0.75)

    iqr = q3 - q1

    lower_bound = q1 -(1.5 * iqr)
    upper_bound = q3 +(1.5 * iqr)

    return (lower_bound,upper_bound)
```

Figure 14 - IQR Method

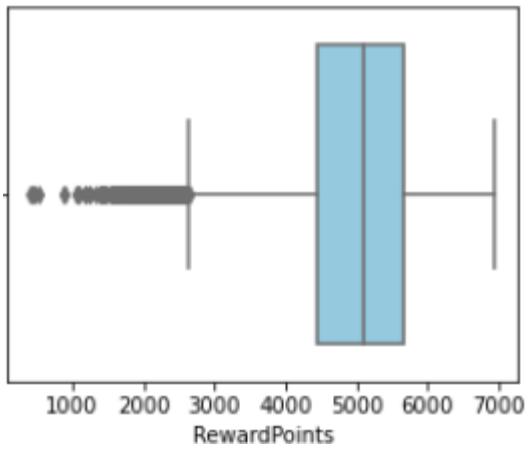

Figure 15 - Boxplot for the identification of outliers in the Numerical Variable "RewardPoints"

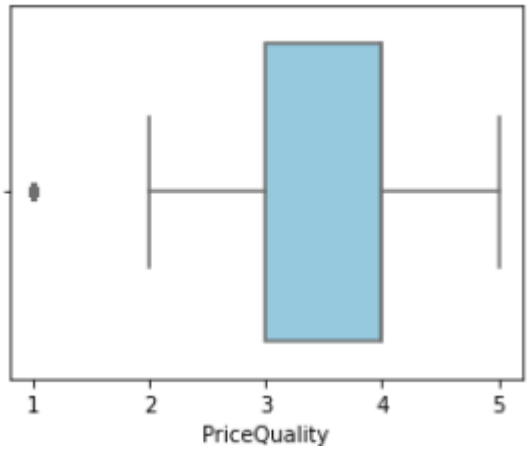

Figure 16 - Boxplot for the identification of outliers in the Numerical Variable "PriceQuality"

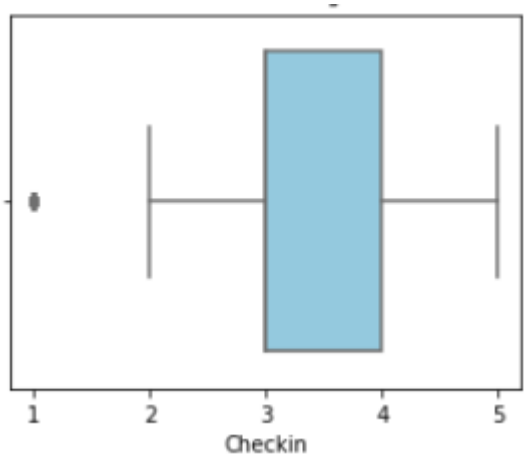

Figure 17 - Boxplot for the identification of outliers in the Numerical Variable "Checkin"

```
straight_lining_entries = df[(df['Comfort'] == df['ReceptionSchedule']) & (df['Comfort'] == df['FoodDrink']) & (df['Comfort']
```

```
straight_lining_entries
```

	Churn	Longevity	Year_Birth	TypeTravel	RoomType	RewardPoints	Comfort	ReceptionSchedule	FoodDrink	Location	...	Amenities	Staff	Onlir
1548	nochurn	yes	1981	business	single	5289	4	4	4	4	...	4	4	
15259	nochurn	yes	1976	business	single	6699	5	5	5	5	...	5	5	

2 rows × 21 columns

Figure 18 - Straight Lining Entries

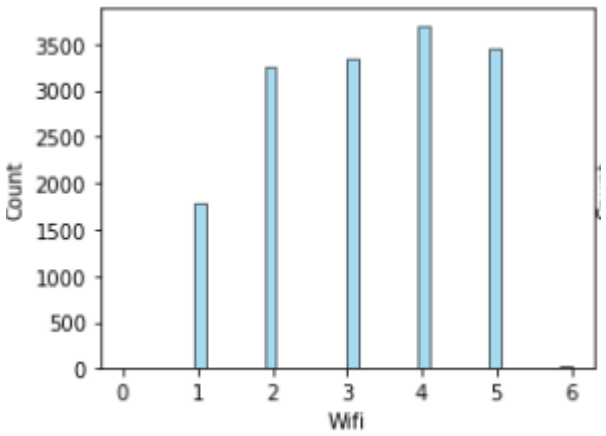

Figure 19 - Histplot for Numerical Variable "Wifi"

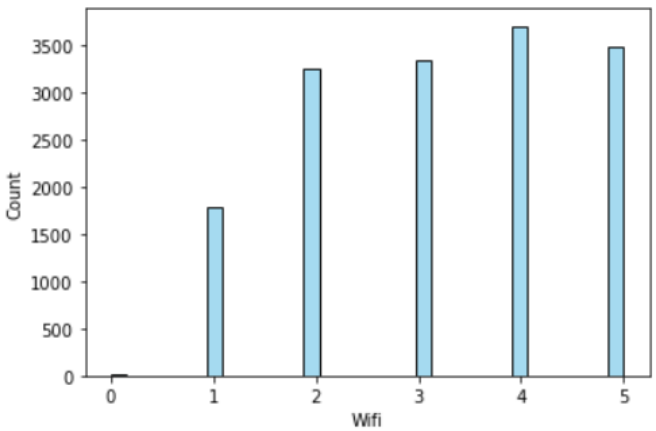

Figure 20 - Histplot for Numerical Transformed Variable "Wifi"

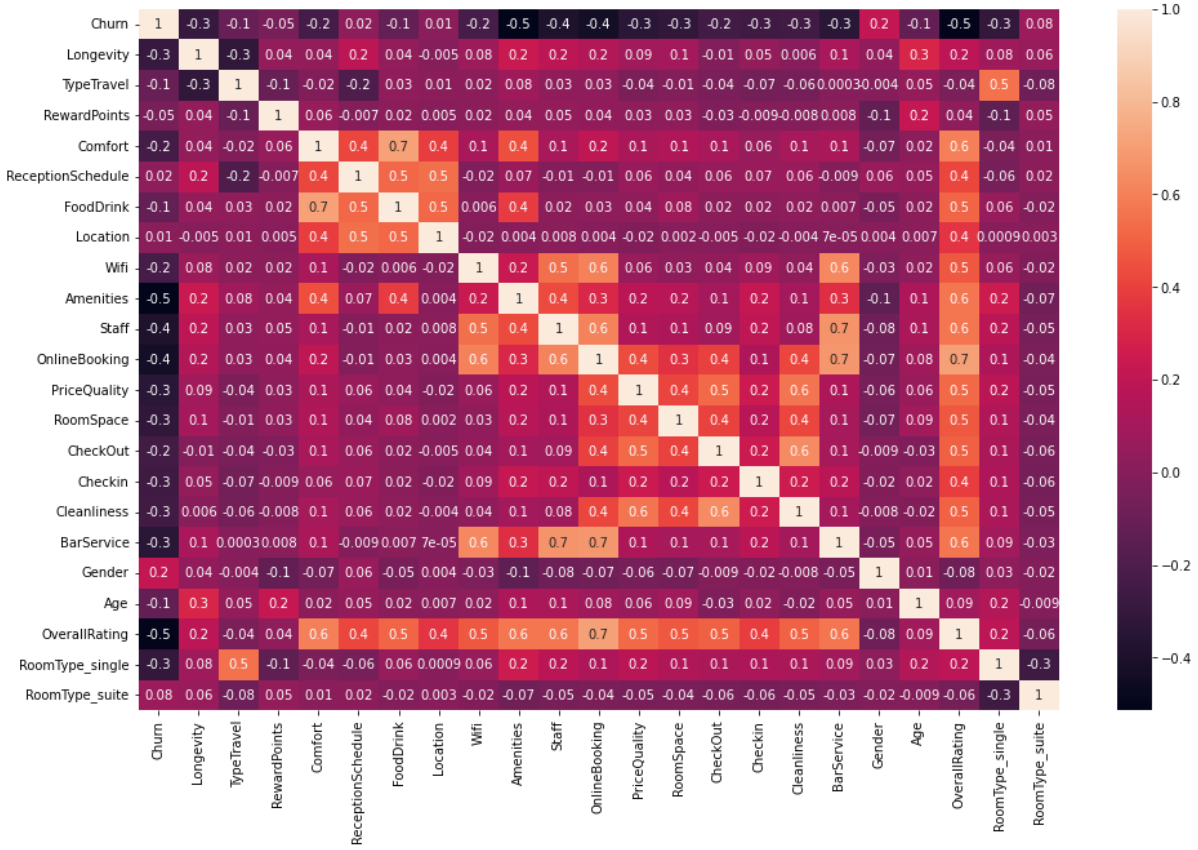

Figure 21 - Correlation Matrix

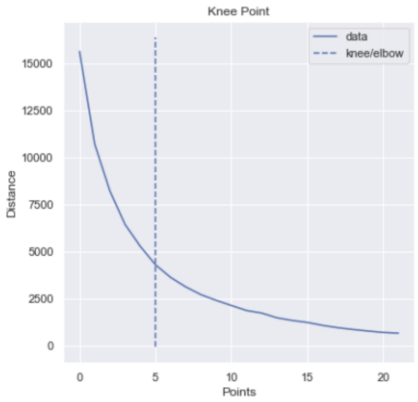

Figure 22 - Kmeans Client Perspective Elbow Method

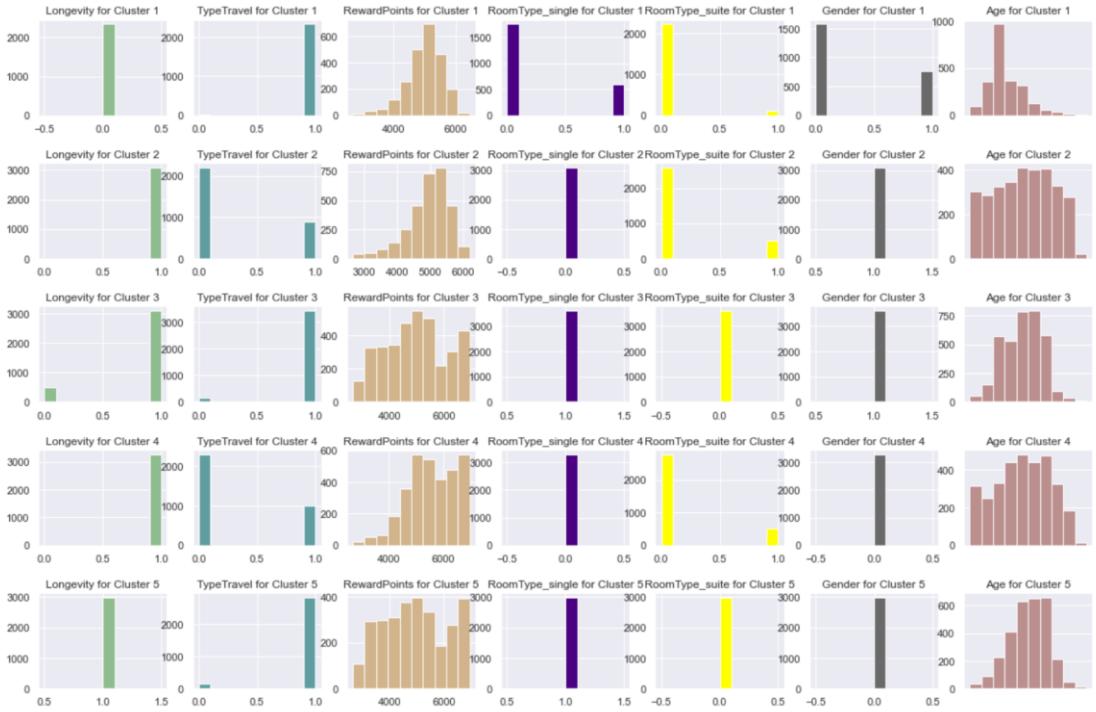

Figure 23 - Kmeans Variable Distribution for Client Clusters

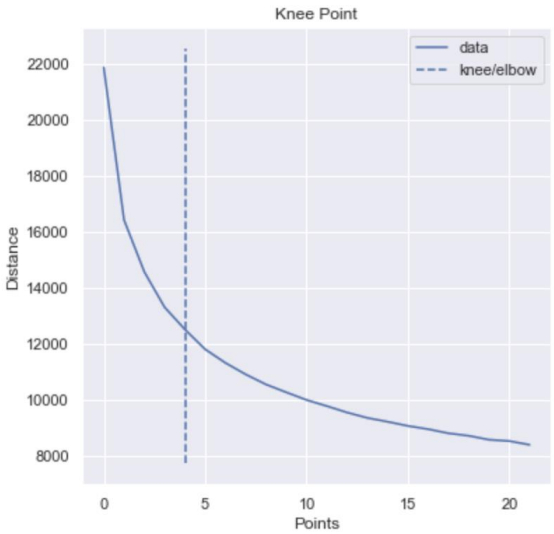

Figure 24 - Kmeans Business Perspective Elbow Method

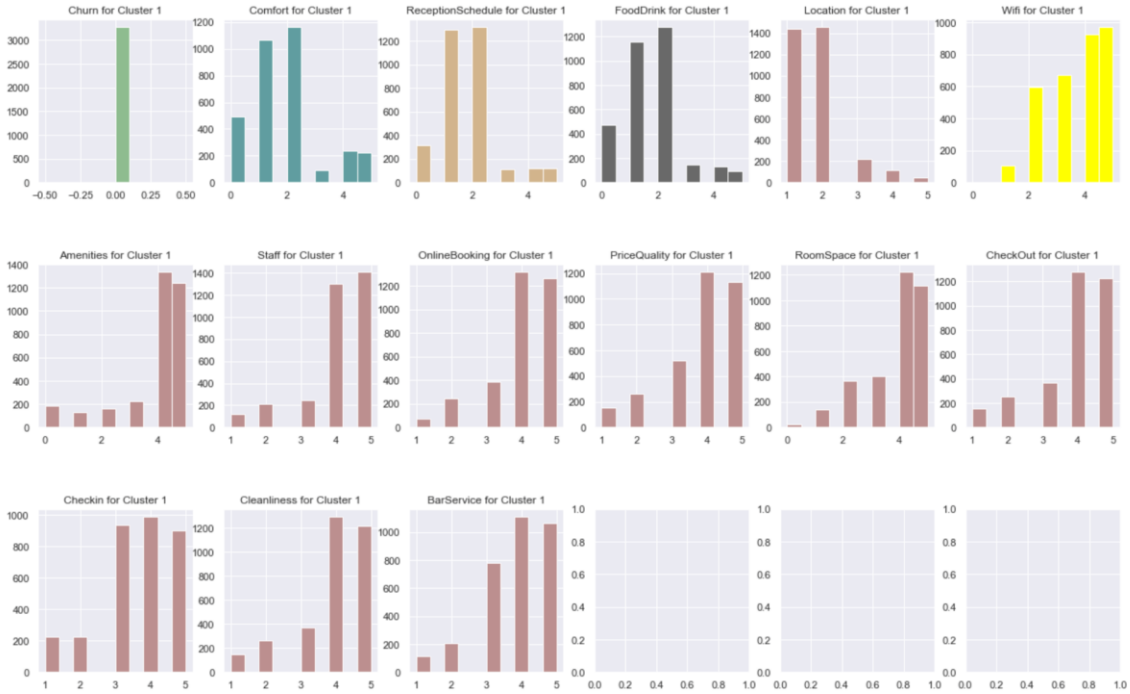

Figure 25 - Kmeans Variable Distribution for Business Clusters Cluster 1

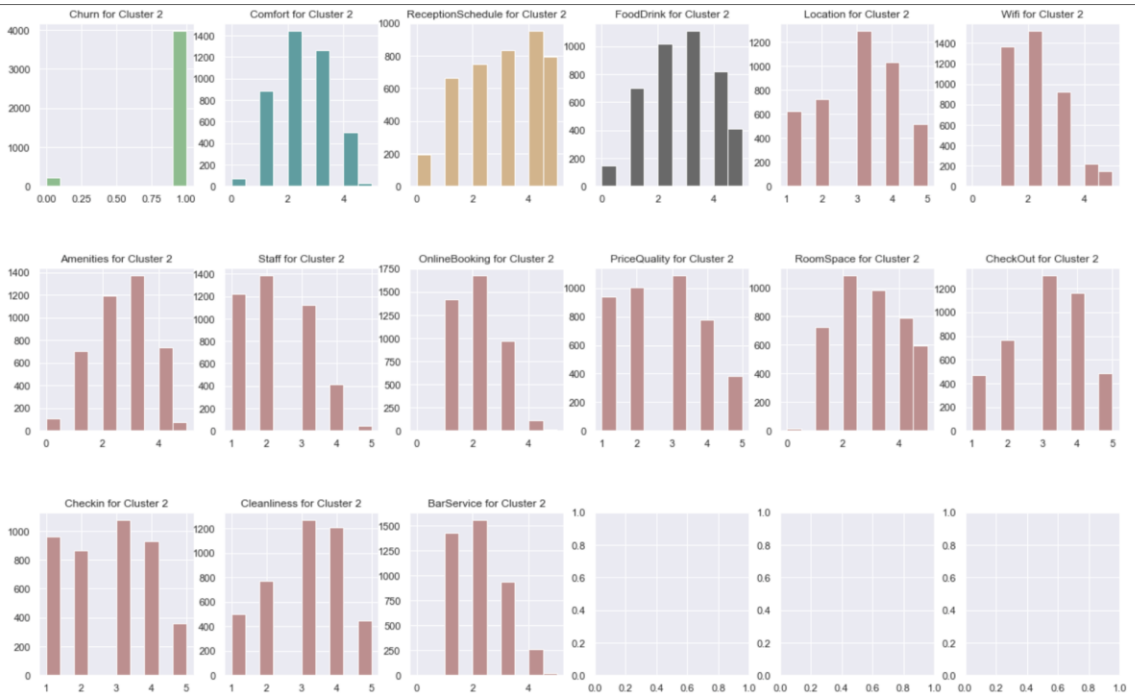

Figure 26 - Kmeans Variable Distribution for Business Clusters Cluster 2

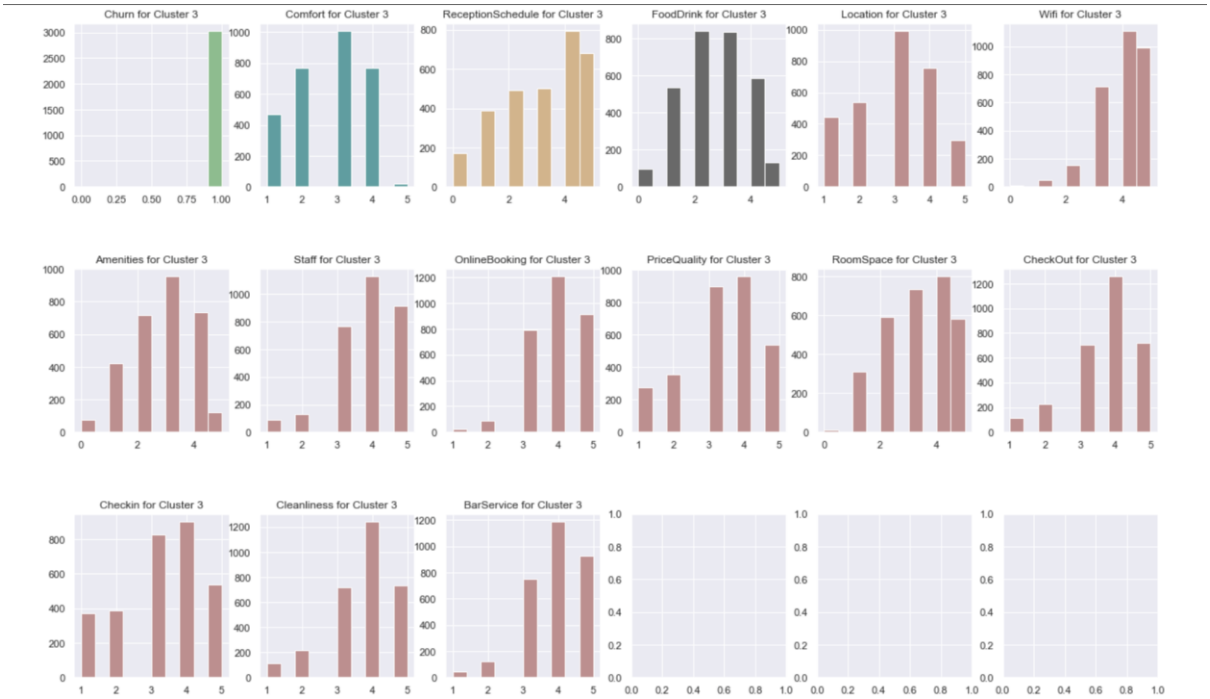

Figure 27 - Kmeans Variable Distribution for Business Clusters Cluster 3

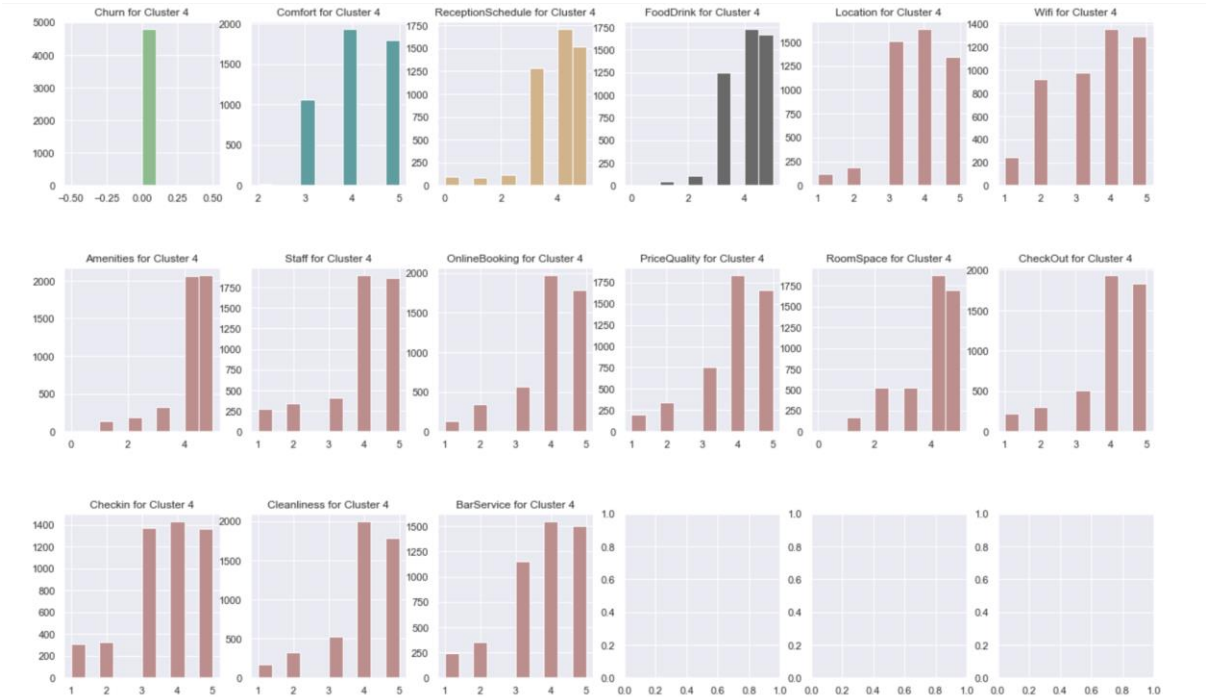

Figure 28 - Kmeans Variable Distribution for Business Clusters Cluster 4

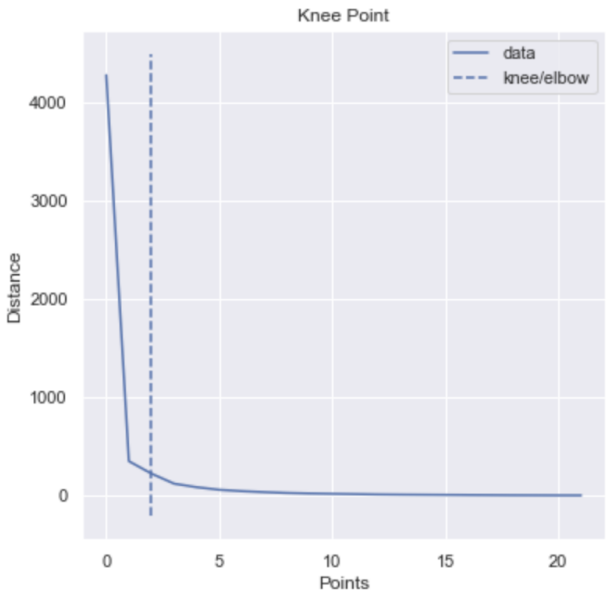

Figure 29 - Kmeans Short Business Perspective Elbow Method

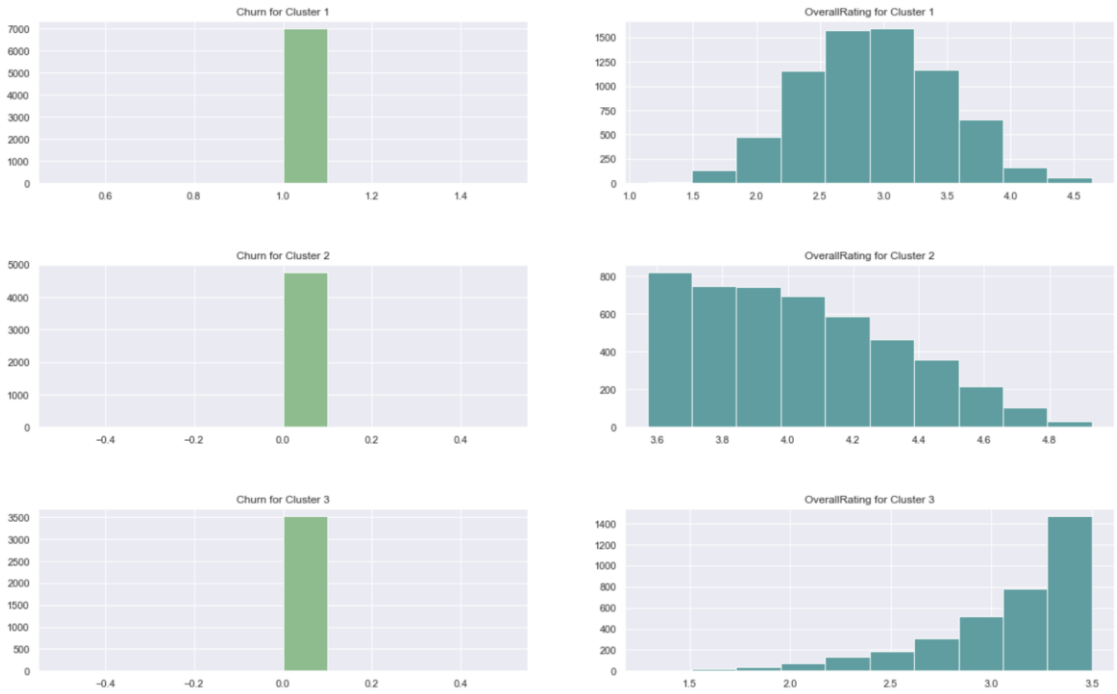

Figure 30 - Kmeans Variable Distribution for Short-Business Clusters Cluster

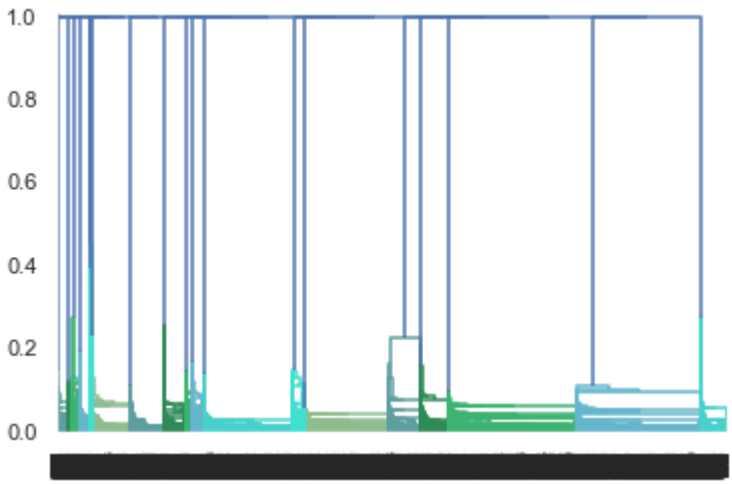

Figure 31 - Dendrogram for the Client perspective with Single Linkage

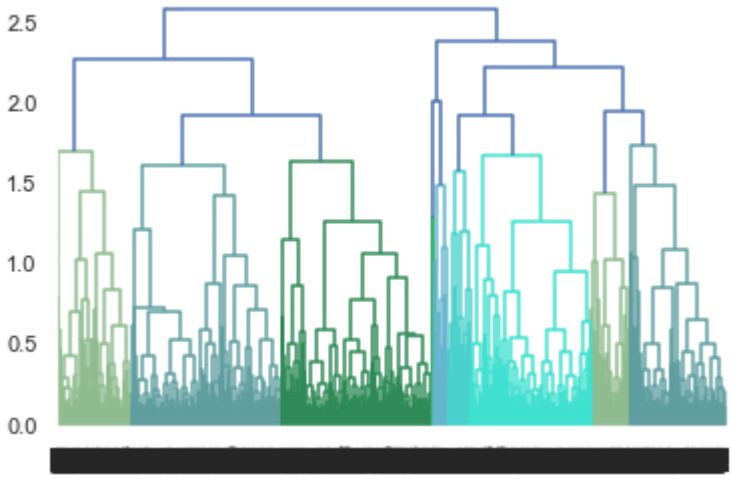

Figure 32 - Dendrogram for the Client perspective with Complete Linkage

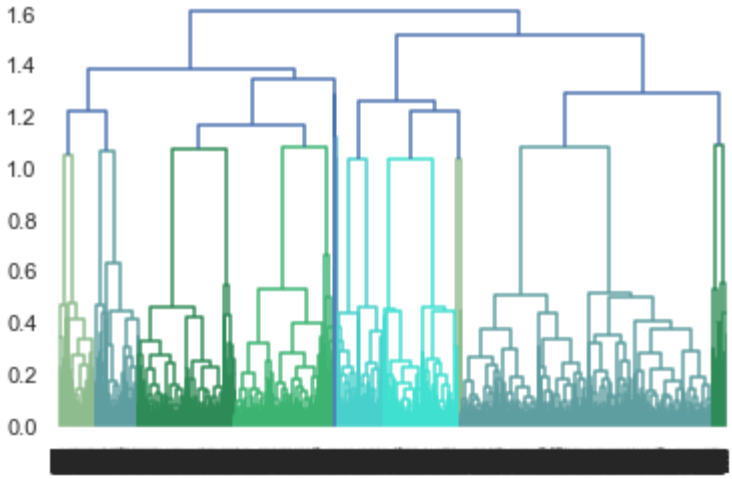

Figure 33 - Dendrogram for the Client perspective with Average Linkage

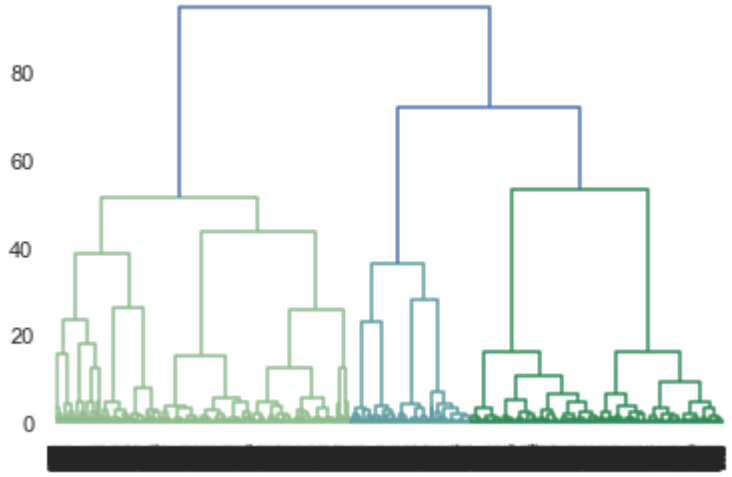

Figure 34 - Dendrogram for the Client perspective with Ward Linkage

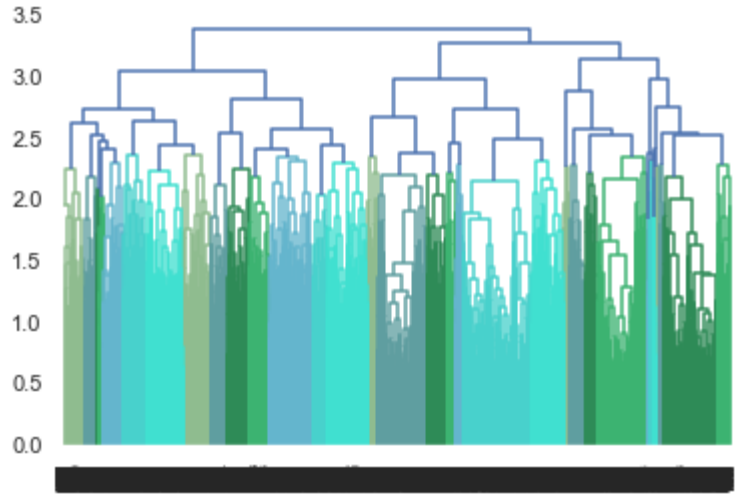

Figure 35 - Dendrogram for the Business perspective with Complete Linkage

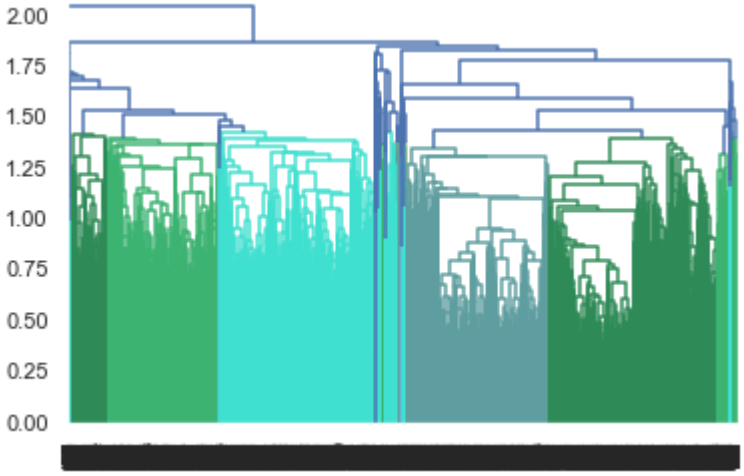

Figure 36 - Dendrogram for the Business perspective with Average Linkage

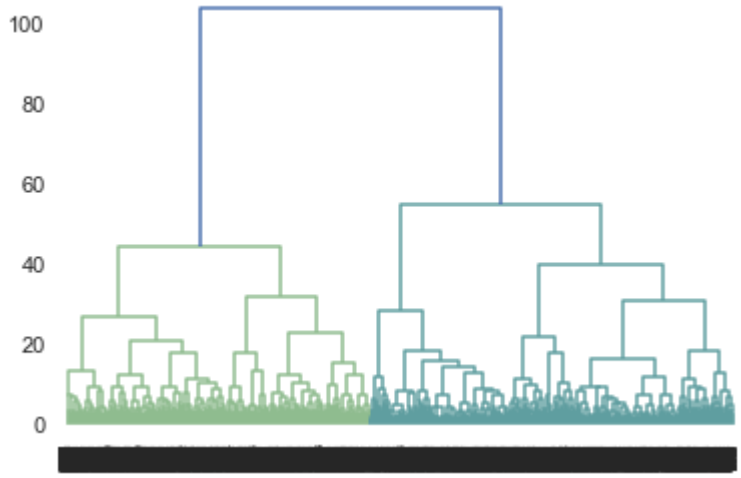

Figure 37 - Dendrogram for the Business perspective with Ward Linkage

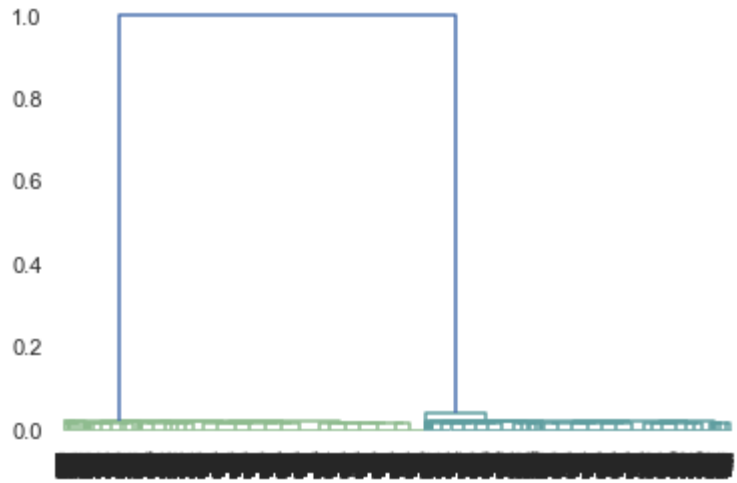

Figure 38 - Dendrogram for the Short-Business perspective with Single Linkage

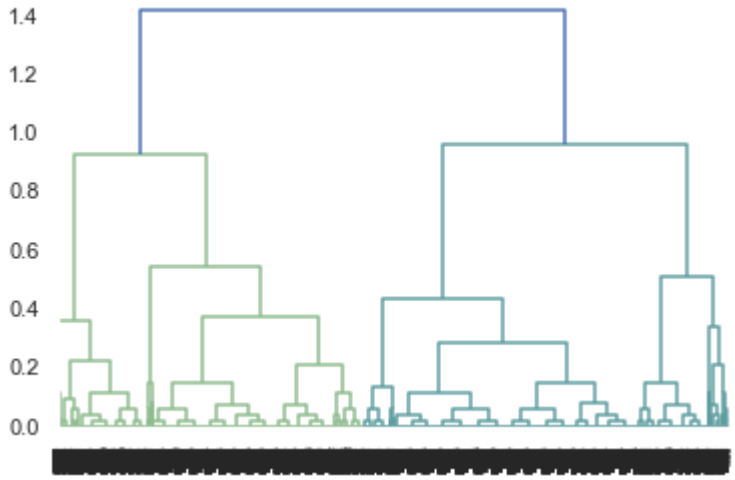

Figure 39 - Dendrogram for the Short-Business perspective with Complete Linkage

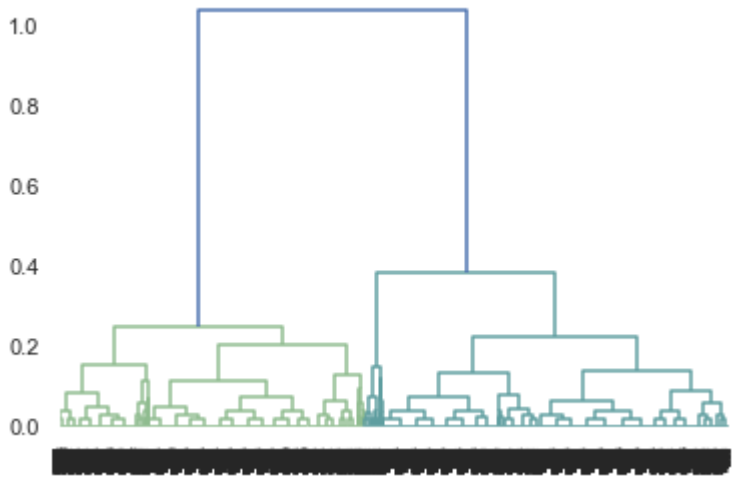

Figure 40 - Dendrogram for the Short-Business perspective with Average Linkage

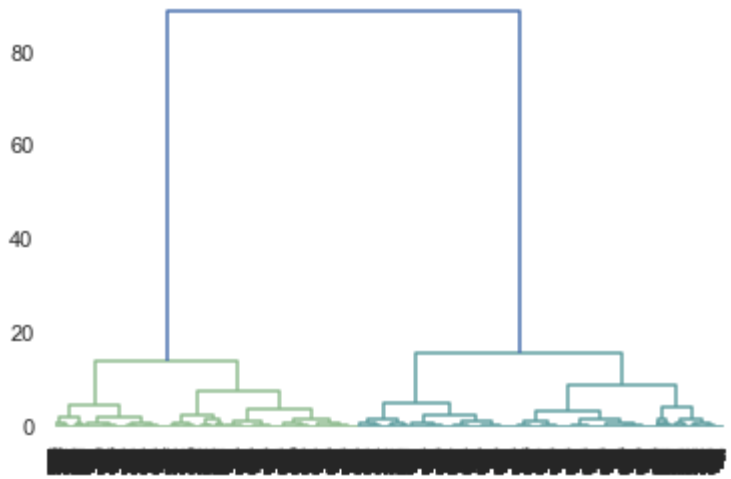

Figure 41 - Dendrogram for the Short-Business perspective with Ward Linkage

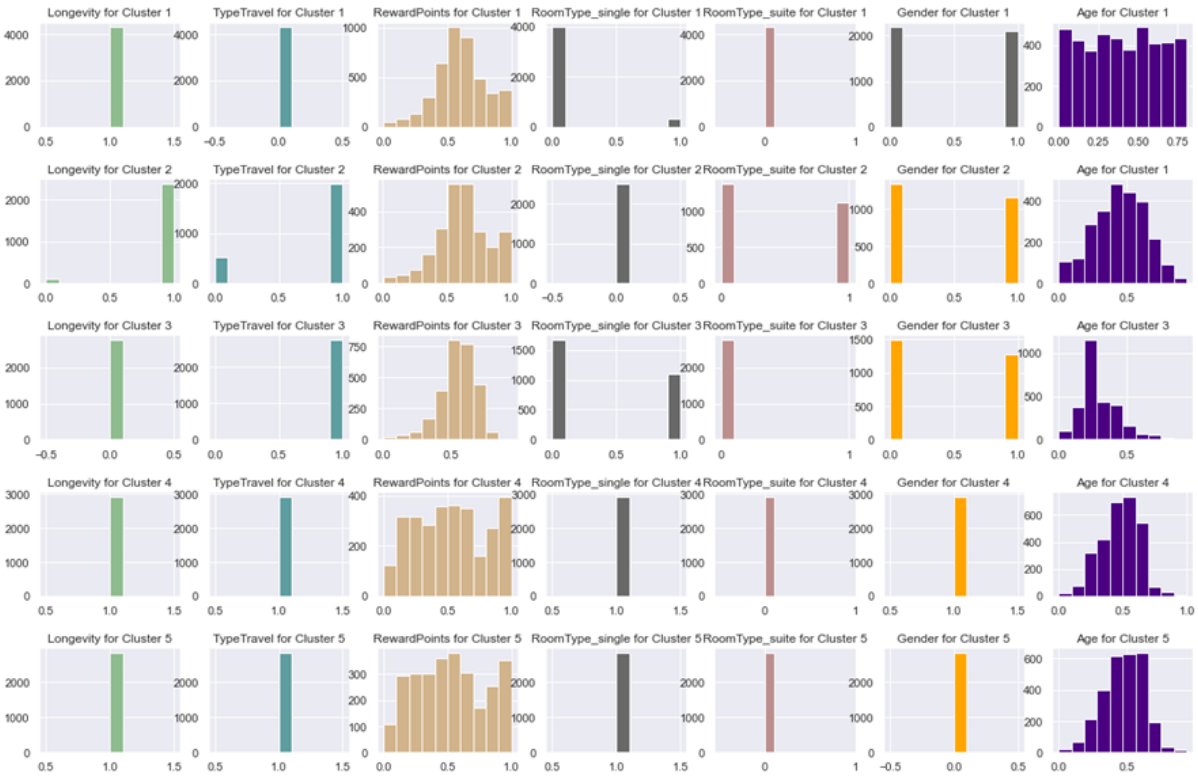

Figure 42 – Hierarchical Clustering Variable Distribution for Client Clusters

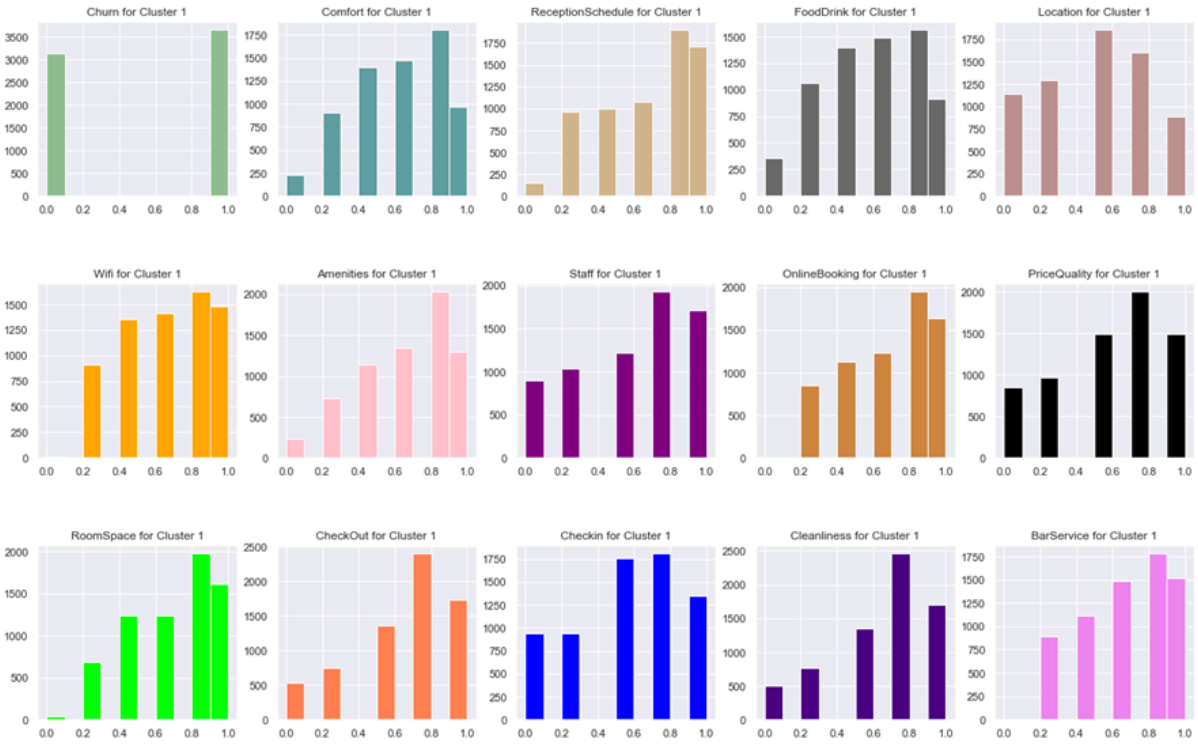

Figure 43 - Hierarchical Clustering Variable Distribution for Client Cluster 1

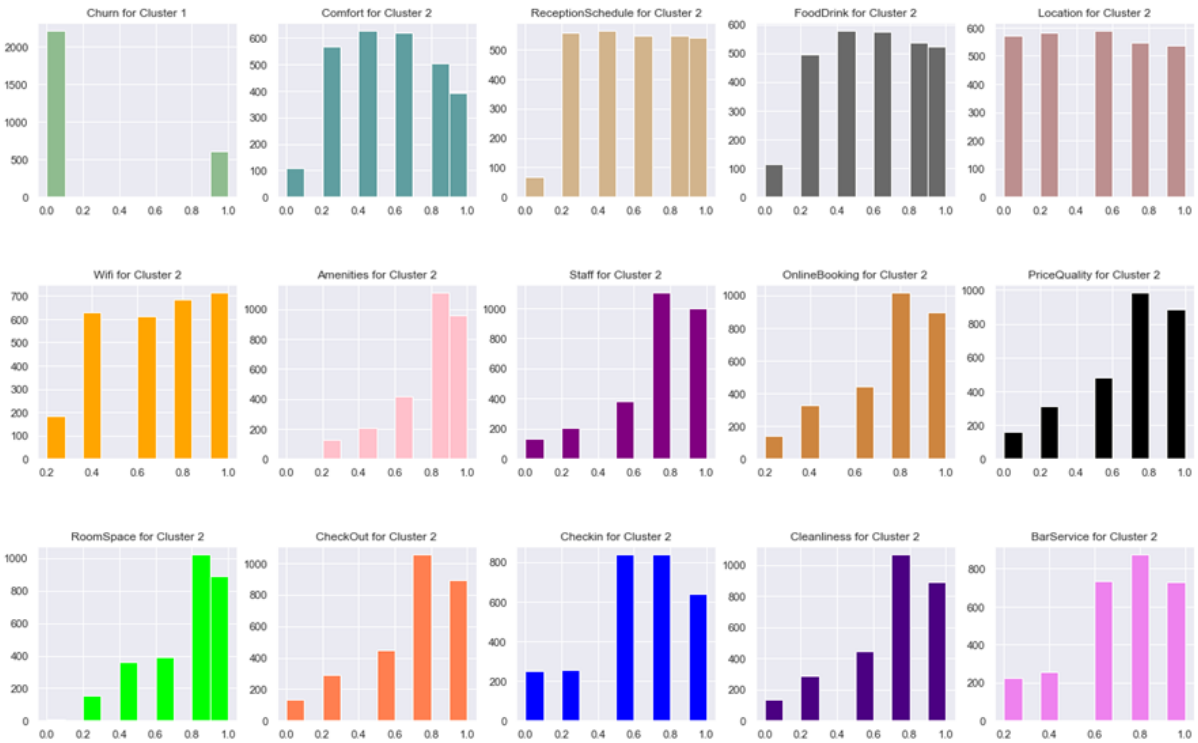

Figure 44 - Hierarchical Clustering Variable Distribution for Client Cluster 2

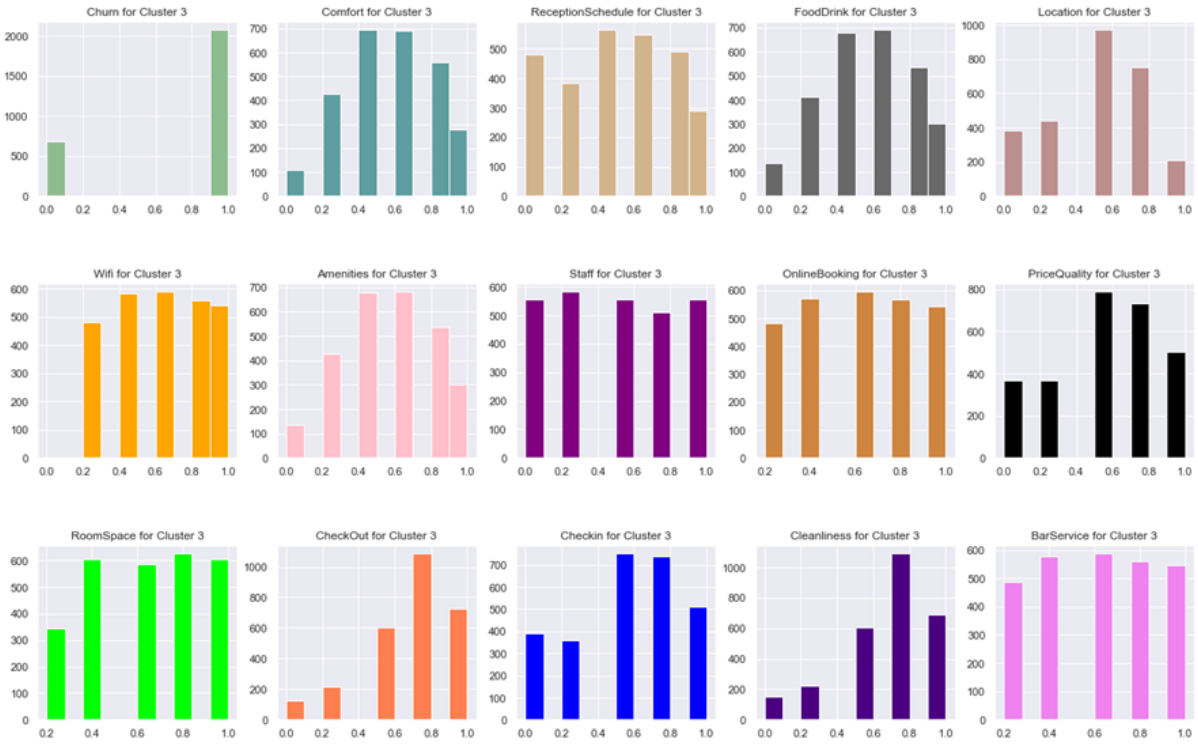

Figure 45 - Hierarchical Clustering Variable Distribution for Client Cluster 3

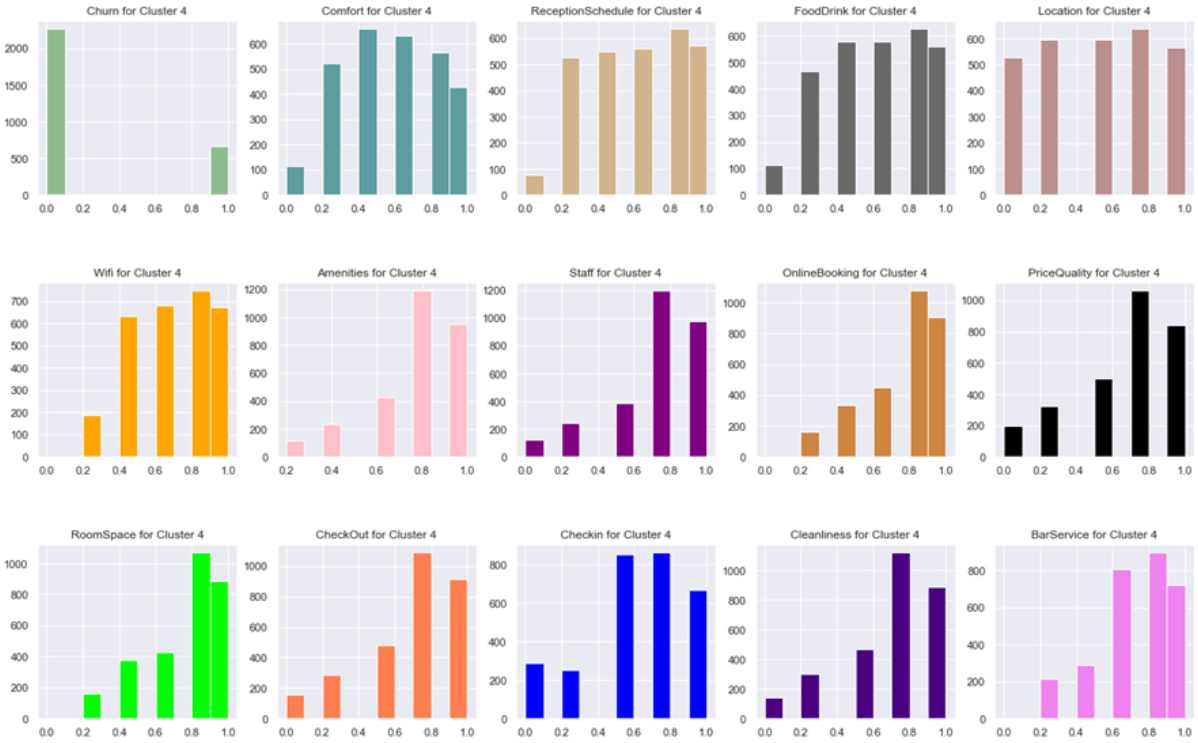

Figure 46 - Hierarchical Clustering Variable Distribution for Client Cluster 4

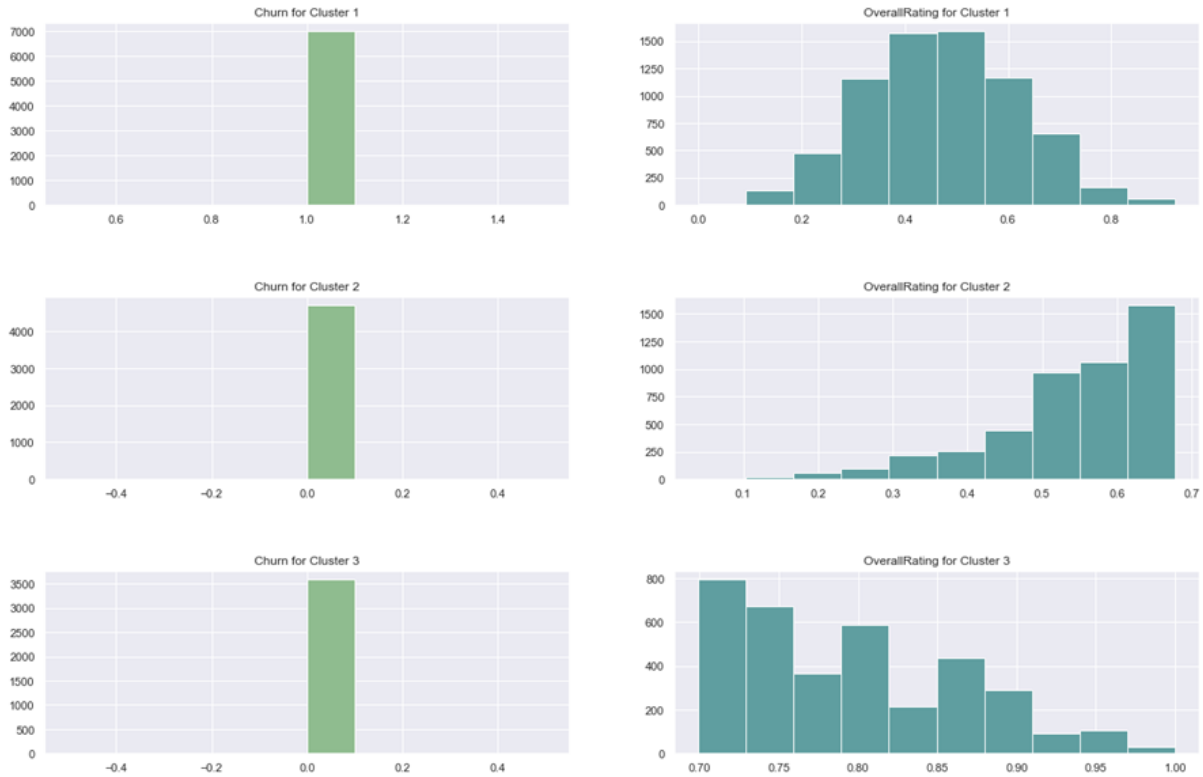

Figure 47 – Hierarchical Clustering Variable Distribution for Short-Business Clusters

```
#Number of observations in each cluster
df['Client'].value_counts()
```

```
2    3604
3    3282
1    3092
4    2975
0    2341
```

Figure 48 – K-Means Customers' Distribution per cluster for Client

```
df_client['Client'].value_counts()
```

```
0    4288
3    2925
4    2824
2    2758
1    2499
```

Figure 49 – Hierarchical Clustering Customers' Distribution per cluster for Client

```
#Number of observations in each cluster
df['Business'].value_counts()
```

```
3    4795
1    4194
0    3277
2    3028
```

Figure 50– K-Means Customers' Distribution per cluster for Business

```
df_business['Business'].value_counts()
```

```
0    6787
3    2925
1    2824
2    2758
```

Figure 51 – Hierarchical Clustering Customers' Distribution per cluster for Business

```
#Number of observations in each cluster
df['Short-Business'].value_counts()
```

```
0    7002
1    4766
2    3526
```

Figure 52– K-Means Customers' Distribution per cluster for Short-Business

```
df_short_business['Short_Business'].value_counts()
```

```
0    7002
1    4700
2    3592
```

Figure 53– Hierarchical Clustering Customers' Distribution per cluster for Short-Business

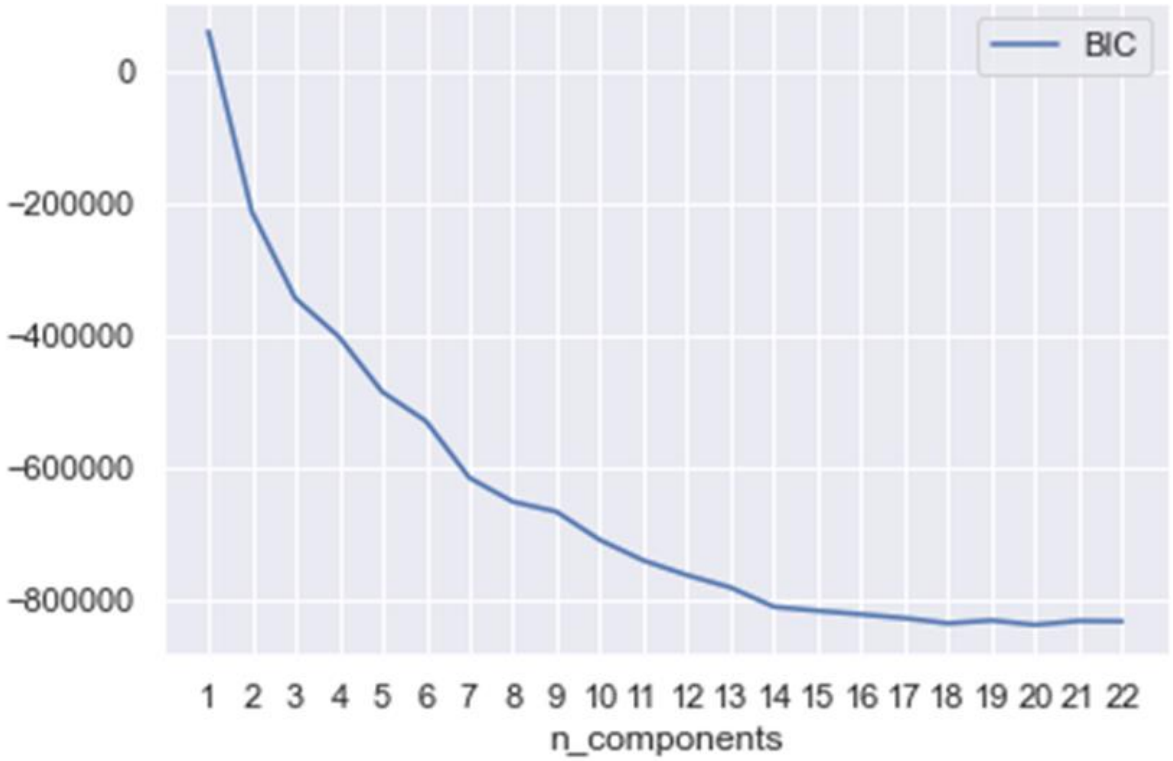

Figure 54 – GMM BIC Criteria to select number of clusters for Client

```
silhouette_score(client, Client)
```

```
0.4272143308028122 4
```

Figure 55 – GMM Silhouette Score for Client

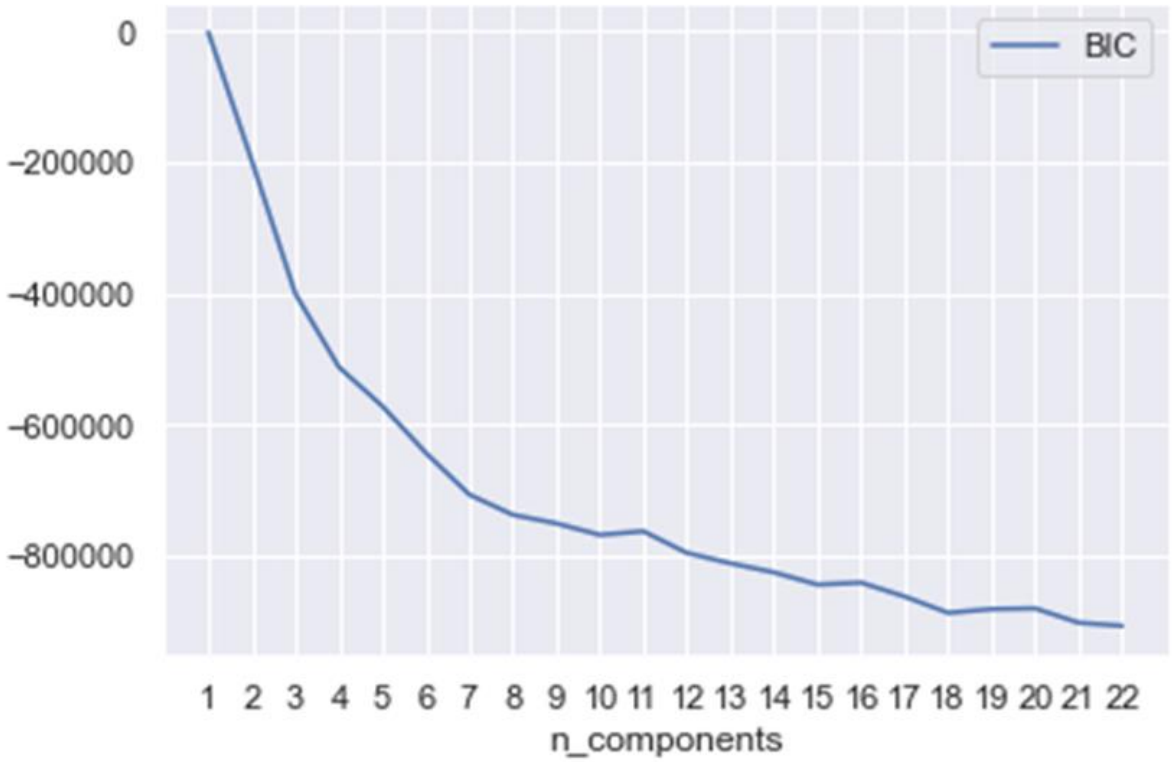

Figure 56 – GMM BIC Criteria to select number of clusters for Business

```
silhouette_score(business, Business)
```

```
0.10436777627521752
```

Figure 57 – GMM Silhouette Score for Business

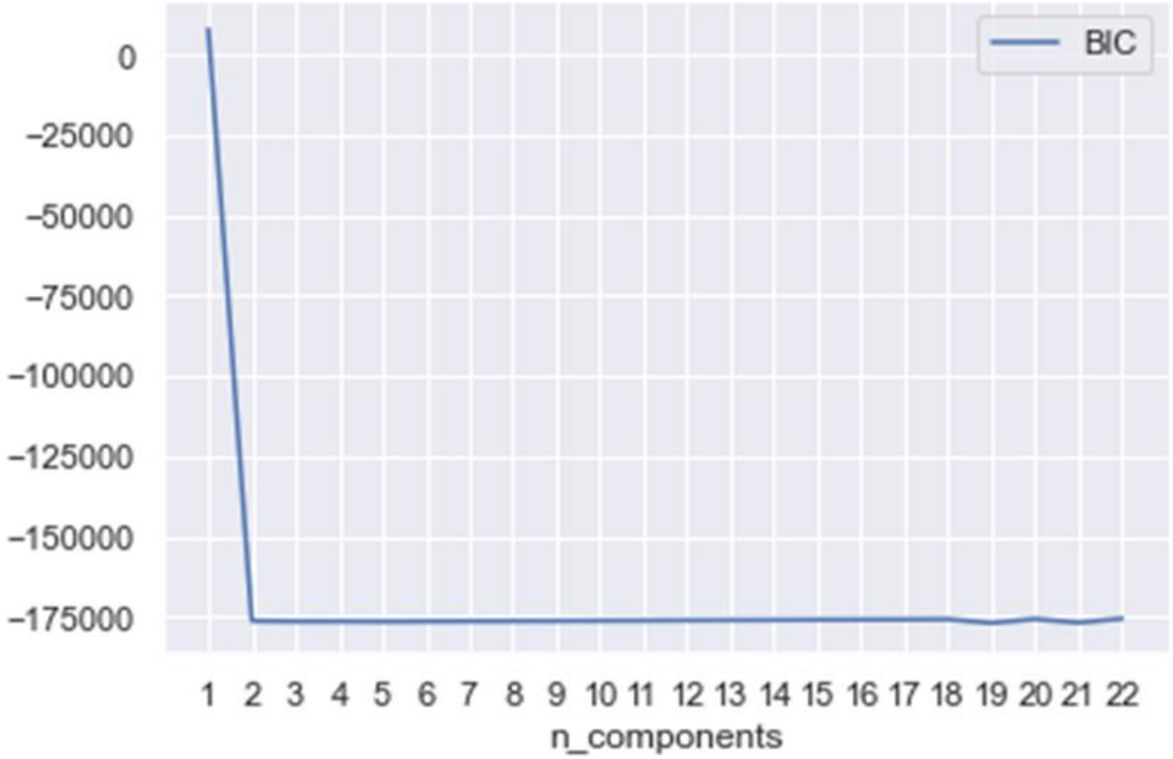

Figure 58 – GMM BIC Criteria to select number of clusters for Short-Business

```
silhouette_score(short_business, Short_Business)
```

```
0.6789677663035634
```

Figure 59 – GMM Silhouette Score for Short-Business

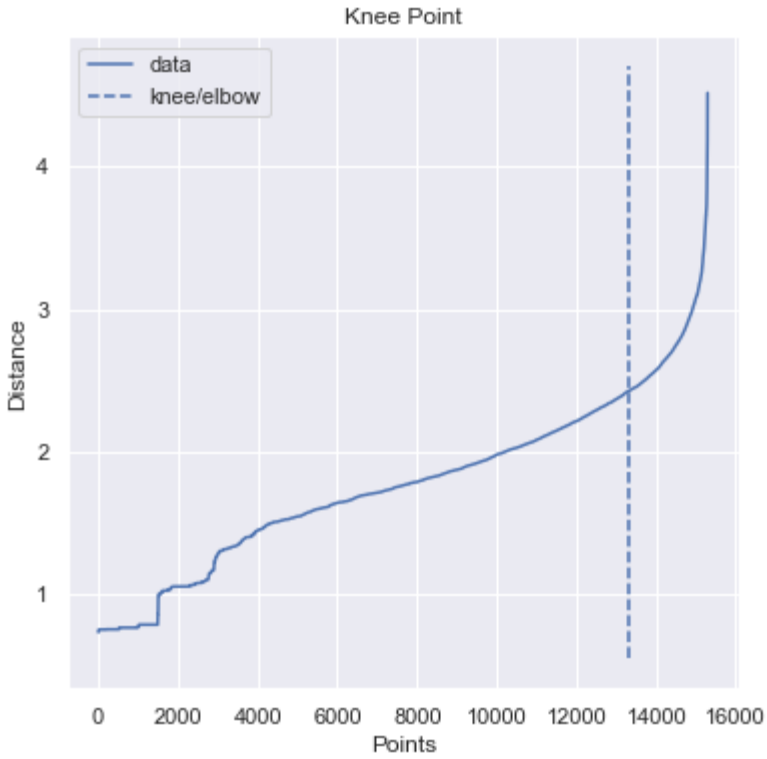

Figure 60 - DBSCAN eps approximation

```
client_df_std_scaled : - eps => 0.022035071756774105
                       - cluster_number => 1
                       - n_noise => 15284
                       - each_cluster => [10]
  ----------------------------------------------------------------------------------------------------------------------
business_df_std_scaled : - eps => 0.7572816238848386
                       - cluster_number => 12
                       - n_noise => 15085
                       - each_cluster => [26, 23, 16, 23, 17, 21, 17, 16, 17, 17, 10, 6]
  ----------------------------------------------------------------------------------------------------------------------
short_business_df_std_scaled : - eps => 1e-07
                       - cluster_number => 93
                       - n_noise => 12
                       - each_cluster => too many to show
  ----------------------------------------------------------------------------------------------------------------------
client_df_norm_scaled : - eps => 0.004872389791183318
                       - cluster_number => 1
                       - n_noise => 15284
                       - each_cluster => [10]
  ----------------------------------------------------------------------------------------------------------------------
business_df_norm_scaled : - eps => 0.19999999999999996
                       - cluster_number => 6
                       - n_noise => 15185
                       - each_cluster => [48, 16, 16, 7, 5, 17]
  ----------------------------------------------------------------------------------------------------------------------
short_business_norm_std_scaled : - eps => 1e-07
                       - cluster_number => 93
                       - n_noise => 12
                       - each_cluster => too many to show
```

Figure 61 - DBSCAN metrics with min_samples equals to number of features plus one

```
client_df_std_scaled : - eps => 0.060858769613947816
                       - cluster_number => 2
                       - n_noise => 15266
                       - each_cluster => [15, 13]
  ----------------------------------------------------------------------------------------------------------------------------
business_df_std_scaled : - eps => 0.7687583236245679
                       - cluster_number => 4
                       - n_noise => 15215
                       - each_cluster => [22, 13, 21, 23]
  ----------------------------------------------------------------------------------------------------------------------------
short_business_df_std_scaled : - eps => 1e-07
                       - cluster_number => 87
                       - n_noise => 43
                       - each_cluster => too many to show
  ----------------------------------------------------------------------------------------------------------------------------
client_df_norm_scaled : - eps => 0.012839394163931348
                       - cluster_number => 2
                       - n_noise => 15263
                       - each_cluster => [18, 13]
  ----------------------------------------------------------------------------------------------------------------------------
business_df_norm_scaled : - eps => 0.20000000000000007
                       - cluster_number => 1
                       - n_noise => 15273
                       - each_cluster => [21]
  ----------------------------------------------------------------------------------------------------------------------------
short_business_norm_std_scaled : - eps => 1e-07
                       - cluster_number => 87
                       - n_noise => 43
                       - each_cluster => too many to show
```

Figure 62- DBSCAN metrics with min_samples equals to number of features plus five

```
client_df_std_scaled : - eps => 0.13652538143097653
                       - cluster_number => 1
                       - n_noise => 15215
                       - each_cluster => [79]
  ----------------------------------------------------------------------------------------------------------------------------
business_df_std_scaled : - eps => 1.1022640187631039
                       - cluster_number => 4
                       - n_noise => 14867
                       - each_cluster => [108, 145, 103, 71]
  ----------------------------------------------------------------------------------------------------------------------------
short_business_df_std_scaled : - eps => 1e-07
                       - cluster_number => 65
                       - n_noise => 593
                       - each_cluster => too many to show
  ----------------------------------------------------------------------------------------------------------------------------
client_df_norm_scaled : - eps => 0.027324331461557255
                       - cluster_number => 1
                       - n_noise => 15220
                       - each_cluster => [74]
  ----------------------------------------------------------------------------------------------------------------------------
business_df_norm_scaled : - eps => 0.3201562118716424
                       - cluster_number => 2
                       - n_noise => 15119
                       - each_cluster => [106, 69]
  ----------------------------------------------------------------------------------------------------------------------------
short_business_norm_std_scaled : - eps => 1e-07
                       - cluster_number => 65
                       - n_noise => 593
                       - each_cluster => too many to show
  ----------------------------------------------------------------------------------------------------------------------------
```

Figure 63 - DBSCAN metrics with min_samples equals to number of features plus fifty

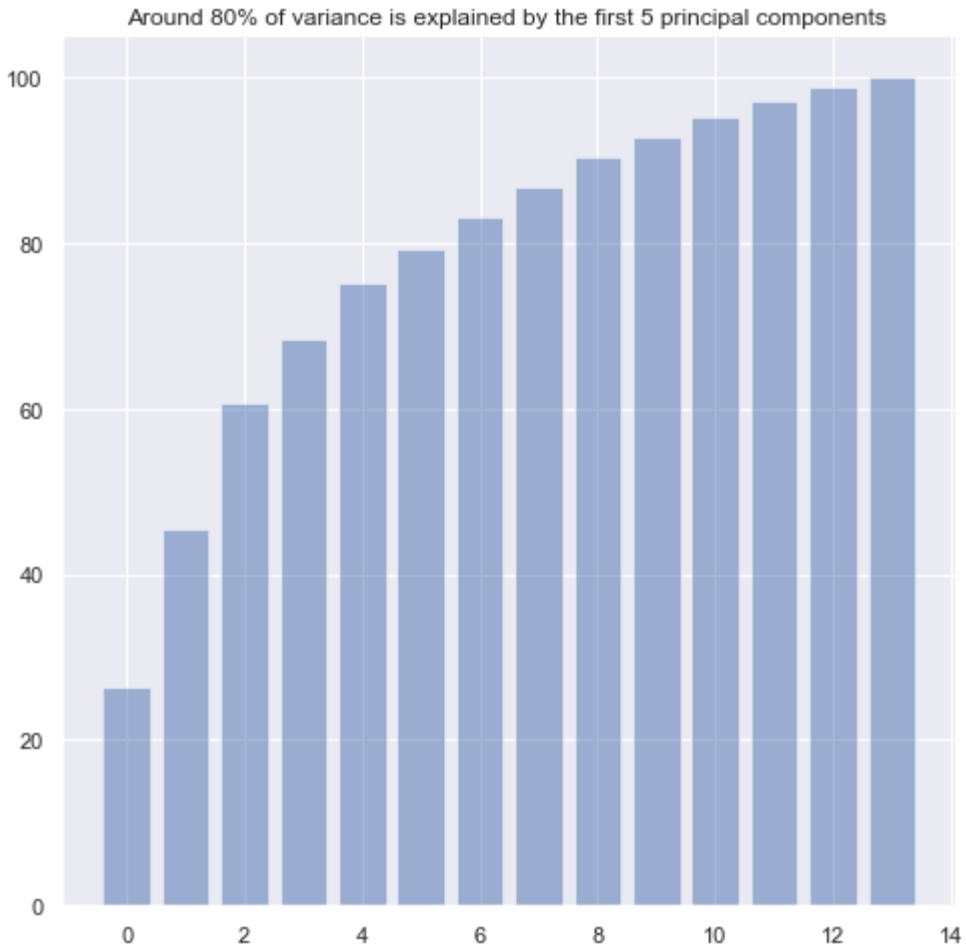

Figure 64 - PCA Variance

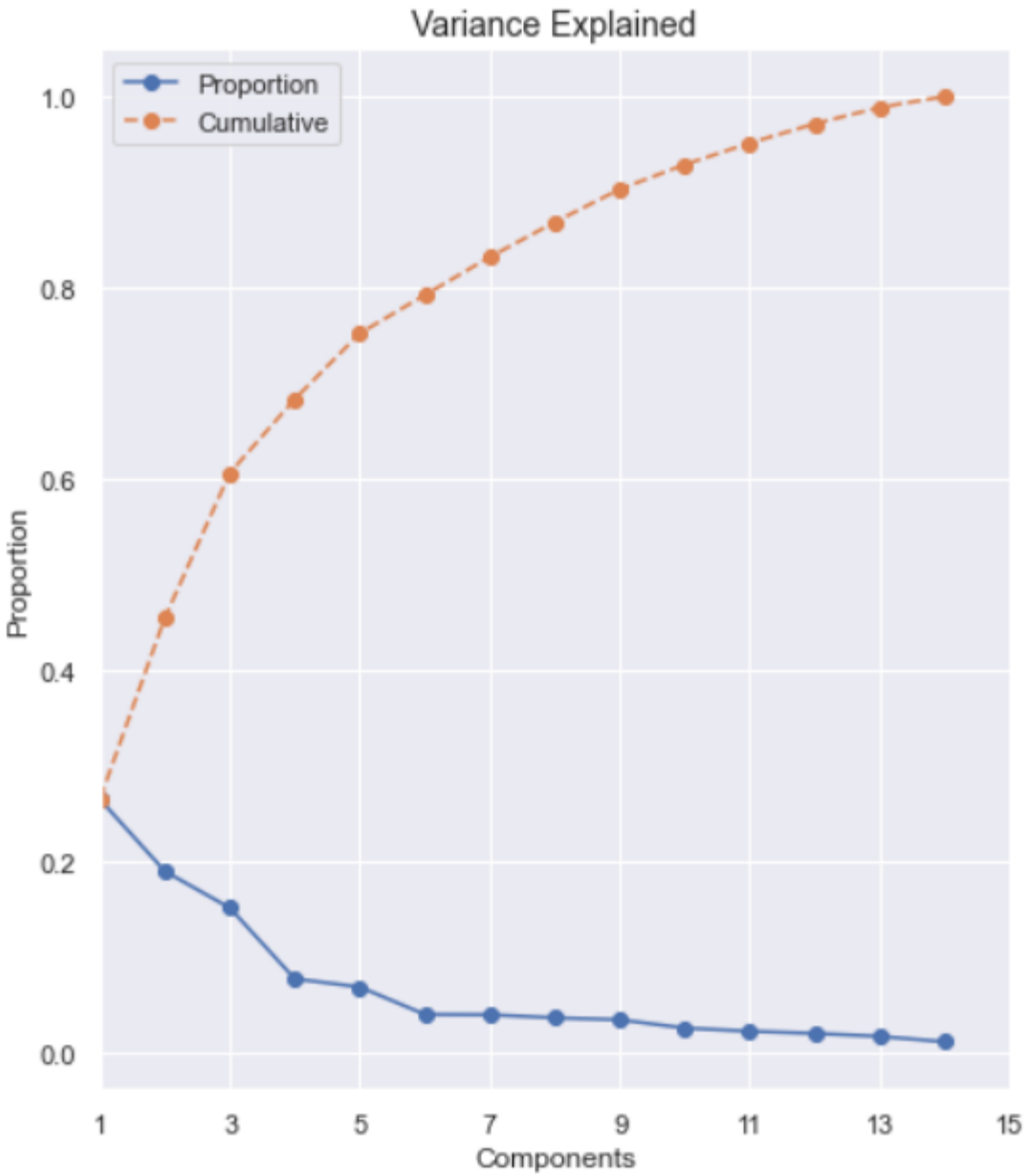

Figure 65 - PCA Variance Explained per Components

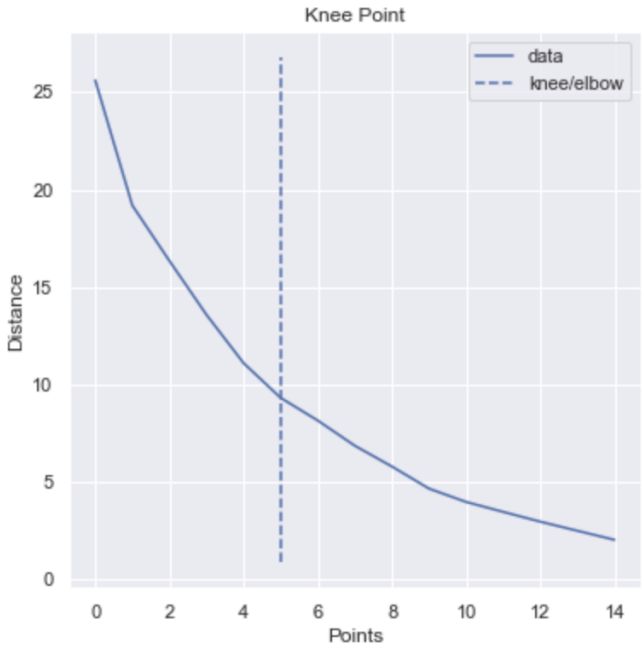

Figure 66 - Kmeans Merged clusters elbow method

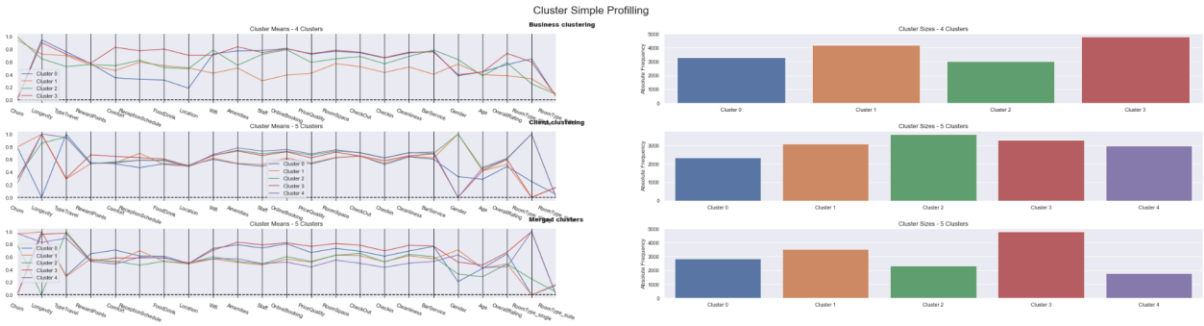

Figure 67 - Kmeans clusters side by side

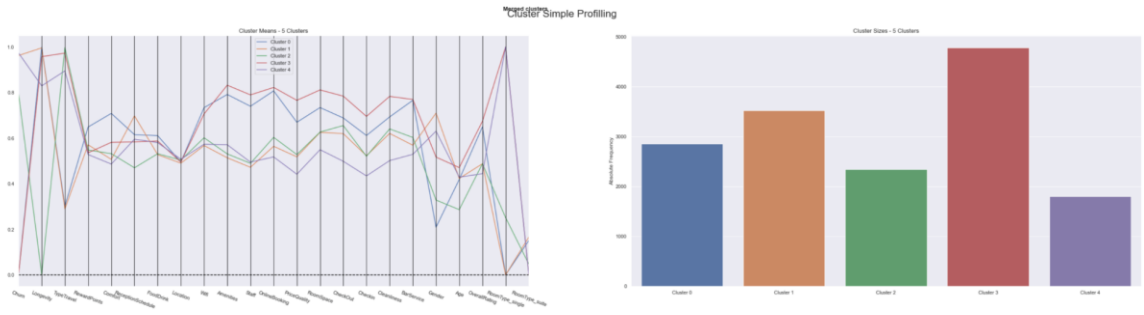

Figure 68 - Kmeans Merged clusters

7. Theoretical Explanation of Self Study Theories

K-Means

The KMeans algorithm is a hard partitioning algorithm method, meaning that every data point in a dataset belongs to one and only one cluster. This algorithm is very effective in finding and creating data groups, with the main objective of minimizing the Sum of Squared Error (SSE), given by the expression below, between clusters.

$$SSE = \sum_{i=1}^{k} \sum_{x \in C} \text{dist}^2 (m_i, x)$$

Where x is an observation of cluster Ci, mi is a representative of the same cluster and k is the number of partitions found beforehand. The simplest way to reduce this error is to increase k, however this is not the optimal solution given that our objective is to minimize the number of clusters. In an extreme example we can say that if the number of clusters is equal to the total number of records (k=n), SSE would be 0, though no useful information would be obtained by it. A balance must be struck by these two variables.

To pursue this model one must set the intended number of clusters (sometimes set to random), followed by assigning each data point to the nearest seed and finally computing the centroids of each of the clusters formed before repeating the process. The iterative process will be finished once centroids no longer show significant changes.

The KMeans model is easily understandable in its core, with the added advantage that it's fast in its implementation and normally it converges into a solution. On the other hand it is a very sensitive model in regards to the initial positioning of the seeds. The presence of outliers is also something that can greatly affect the results, not to mention that the KMeans is not effective at finding clusters with shapes other than spherical. Other disadvantages are that it may not be appropriate for larger datasets due to the Euclidean distance used, and the fact that it may not always be easy to determine the initial number of clusters to use. To overcome this last obstacle we might use the previously described SSE and compute it graphically, which will originate the well known "Elbow Method" that allows to observe and decide the optimal number to assign to k.

There is no concrete way of saying if a cluster model is good, we can however check the intra-cluster and inter-cluster distances. Good clusters should be compact and distant from each other

GMM Theory

Gaussian Mixture Models (GMM) can be, in certain situations more appropriate than methods such as KMeans clustering due to the fact that it can oblige to different sized clusters with different correlation

structures between them. Like the aforementioned models, GMM requires the number of clusters to be specified in order to fit the model and find the number of its components.

A better way to describe the GM model would be that by having three peaks in a dataset, each point will have Gaussian Distribution, which will have means and variances associated to them, GMM will identify the probability of each data point belonging to each of these distributions.

Covariance structure should be considered for the application of this model, this may include diagonal, full covariance matrices, or even if all the components will have the same one. Initial conditions need to be specified, and finally implementing regularization of the data.

DBSCAN Theory

DBSCAN means Density Based Spatial Clustering Application with Noise. An unsupervised learning algorithm that usually thrives when other more popular algorithms like k-means have some difficulty, in complex shaped data. It identifies high density areas in the feature space and assigns clusters to these areas.

To do so DBSCAN needs as input parameters the eps (distance used to specify neighbors) and minPts (minimum number of points in a cluster).

Some other important concepts to understand DBSCAN are core points (points with the minimum of minPts in eps radius around itself), border points (points reachable by a core point but with less then minPts in it's surrounding area), and the outliers or noise point (that are not within range of any core point).

A start point is chosen at random, and it's labeled a core point, or noise. If it's a core point then all points in its eps radius become part of this cluster, then those points are classified as core or border points, if they are core points it's neighbor is also added to this cluster and this process continues until this cluster is finished. After that another point is chosen at random with the ones that are not yet classified, and the process starts again for other cluster and it runs utils all the points are classified.

By doing that high density regions are labeled as clusters and clusters that are close to one another are labeled parts of the same cluster, another good output of this algorithm is the noise points or outliers, by identifying that points this method is not sensible to them and can be used for anomaly detection as well.

K Nearest Neighbor Imputer (KNN) Theory

The KNN algorithm is often used as a solution for real life datasets that couldn't be worked on by most statistical and machine learning algorithms due to the fact that they are not complete. This methodology is applied directly in the clustering model with the objective of predicting these missing values based on

data points previously known in the dataset. In this case the, k refers to the number of neighbors needed to forecast unseen data points. KNN allows to compute the distance from these neighbors and the missing values.

The value of k has no theoretical approach to it, however the performace of the model can be evaluated based on the errors of the scoring test. As the value of k is increased gradually it is possible to observe and choose the one with the lowest error. Usually, the number of neighbors increasing means that the error will stabilize soon, which will lead to model convergence.

The most common formula to reach the distance in the KNN imputer is known as the Euclidean Distance, found below (D). This distance can be calculated by the summation of square differences between x and y, being x the vector with known data points, and y the vector to predict values, and square root that sum.

$$D(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

This formula is sensitive to the magnitude of the data, meaning that a greater magnitude can affect the distance measures. The Euclidean Distance can also be affected by skewness distributions, which means that the dataset should be normalized beforehand.

Dimensionality is something that will have an impact on the KNN, this has no obvious solution other than keeping it low, removing irrelevant variables or highly correlated ones. Another particularity of this algorithm is that it requires all data to be numerical so before it being applied, the creation of dummy variables in place of categorical is strictly necessary.

Principal Component Analysis (PCA) Theory

The Principal Component Analysis, or PCA, is a method that allows for the reduction of the number of dimensions. It does this by combining variables that have a high correlation between each other and turning them to variables that contain more information.

The Principal Components (PCs) in this model identify the direction to which the data spread is maximized. The first component will therefore indicate the direction where the variation is largest (PC1). With two dimensions, the second component (PC2) is orthogonal to PC1, which means that there is no correlation between the main components of the variables. Similarly, when there are more than two dimensions the PCA allows for the identification of orthogonal directions which represent the biggest variances in the data. This process is not iterative, meaning that once the algorithm identifies one component , it computes all of them at the same time, and the number of PCs will be similar to the number of input variables. In spite of this fact, the main objective of the PCA is to reduce the data dimensions, so not all of the principal components will be kept. It is important to define some Criterion in order to determine which components are meant to stay.

Some examples of this would be:

Eigenvalue Criterion: Eigenvalue represents the number of inputs entered in the PCA. An example could be, when given an eigenvalue of 1, the principal component will explain the equivalent to 1 variable, meaning that this eigenvalue criteria will retain only principal components with eigenvalues greater than one.

The Proportion of Variance Explained Criterion: Firstly, this requires the specification of how much of the total variability should be explained, then selection of the corresponding principal components can occur.

The Scree Plot Criterion: To apply this criterion one mustindentify a knee in the scree plot made from the cumulative percentage variance explained versus the principal component number must be identified in order to determine the point of diminishing returns in keeping another principal component in the model.

Hierarchical Clustering Theory

Clustering techniques can generally be classified as hierarchical or non-hierarchical. In this particular case, the hierarchical technique can be divided into divisive and agglomerative methods, which will both create a dendrogram according to the category assigned.

When using divisive clustering, the dendrogram that will be built, will use the recursive partitioning of the existing clusters., meaning that all observations will begin by being one single cluster and after every iteration the most distant observations will be separated into a different cluster. This will lead to the final number of observations in the dataset being the same as the final number of clusters.

When taking the alternative approach, agglomerative clustering, the dendrogram in question will be built by combining already existing clusters. With this method, at first every observation is part of a single cluster and after every iteration the two clusters that are closest to each other will be combined into a new one. This happens until the end, where only one aggregated cluster could be found.

To measure the distance between clusters the aforementioned Euclidean distance metric can be used. The other criterion to be aware of is how this distance will be computed, there are several ways of measuring distance (approached below) since these clusters are composed of so many different observations.

Complete Linkage, also known as the farthest neighbor approach, implies that the similarity between clusters is evaluated by the most distant observations from each cluster.

Single Linkage, or nearest neighbor approach, is the opposite of the above, meaning that the similarity between clusters will be measured by the nearest observations in each other.

Average Linkage, as the name implies, will be using the average distance between all records in two clusters. This is useful to reduce the dependence of extreme values.

Ward's Method, known as the minimum variance method, takes into account the loss of information when observations are clustered together. It uses the Error of Sum Squares that, as mentioned above, measures the difference between every observation and the cluster's mean.

For Hierarchical Clustering the iterative steps are clear, a distance metric should be chosen in order to compute the distance between data points. Followed by the selection of a linkage criterion for the distances between clusters. Computing the cluster hierarchy should be the next step. This ends only when only one agglomerative cluster exists or when there are n clusters corresponding to n data points (this will depend on whether the agglomerative or the divisive method was chosen). Finally, after dendrogram observation, the final number of clusters should be chosen.

This method of clustering can be computationally intensive, especially for big data sets, however it is more efficient than other partial algorithms by being "greedy", meaning that it cannot get back after every iteration.

Skewness

In probability theory and statistics, skewness is a measure of the asymmetry of the probability distribution of a random variable about its mean. The skewness value can be positive, zero, negative, or undefined.

For a unimodal distribution, negative skew commonly indicates that the tail is on the left side of the distribution, and positive skew indicates that the tail is on the right. In cases where one tail is long but the other tail is fat, skewness does not obey a simple rule.

If skewness is less than −1 or greater than +1, the distribution can be called highly skewed. If skewness is between −1 and −½ or between +½ and +1, the distribution can be called moderately skewed. If skewness is between −½ and +½, the distribution can be called approximately symmetric.

If one tail is longer than another, the distribution is skewed. These distributions are sometimes called asymmetric as they don't show any kind of symmetry. Symmetry means that one half of the distribution is a mirror image of the other half. For example, the **normal distribution is a symmetric distribution with no skew**. The tails are exactly the same.

A left-skewed distribution has a long left tail. Left-skewed distributions are also called negatively-skewed distributions. That's because there is a long tail in the negative direction on the number line. The mean is also to the left of the peak. We can consider

A right-skewed distribution has a long right tail. Right-skewed distributions are also called positive-skew distributions. That's because there is a long tail in the positive direction on the number line. The mean is also to the right of the peak.

Kurtosis

Kurtosis is a statistical measure that defines how heavily the tails of a distribution differ from the tails of a normal distribution. In other words, kurtosis identifies whether the tails of a given distribution contain extreme values. Along with skewness, kurtosis is an important descriptive statistic of data distribution. However, the two concepts must not be confused with each other. Skewness essentially measures the symmetry of the distribution, while kurtosis determines the heaviness of the distribution tails.

There are three type of kurtosis:

Mesokurtic

Data that follows a mesokurtic distribution shows an excess kurtosis of zero or close to zero. This means that if the data follows a normal distribution, it follows a mesokurtic distribution.

Leptokurtic

Leptokurtic indicates a positive excess kurtosis. The leptokurtic distribution shows heavy tails on either side, indicating large outliers.

Platykurtic

A platykurtic distribution shows a negative excess kurtosis. The kurtosis reveals a distribution with flat tails. The flat tails indicate the small outliers in a distribution.

8. Bibliography

BrownMath. 2022. “Measures of Shape: Skewness and Kurtosis” Accessed 7 April: https://brownmath.com/stat/shape.htm

Larose, D.T and Larose, C.D. Data Mining and Predictive Analytics; 2015; Wiley

Machine Learning Knowledge. 2021. “Tutorial for DBSCAN Clustering in Python” Accessed 12 April: https://machinelearningknowledge.ai/tutorial-for-dbscan-clustering-in-python-sklearn/

Patel, A. A. (2019), Hands-on unsupervised learning using Python: How to build applied machine learning solutions from unlabeled data; O’Reilly

Towards Data Science. 2020. “Silhouette Coefficient” Accessed 9 April: https://towardsdatascience.com/silhouette-coefficient-validating-clustering-techniques-e976bb81d10c#:~:text=Silhouette%20Coefficient%20or%20silhouette%20score%20is%20a%20metric%20used%20to,each%20other%20and%20clearly%20distinguished.

Towards Data Science. 2018. “Gaussian Mixture Model clustering: how to select the number of componentes (clusters)” Accessed 14 April:

https://towardsdatascience.com/gaussian-mixture-model-clusterization-how-to-select-the-number-of-components-clusters-553bef45f6e4