Master's in Data Science and Advanced Analytics, with specialization in Data Science

Data Mining

2021/2022

Project Report

Nuno Penim - m20210998

Paulo Oliveira – m20211002

Gonçalo Gomes – m20211007

5th of January of 2022

Index

1. Introduction

In this project, we are asked to develop a Costumer Segmentation in a way that makes it possible for the marketing department of a fictional insurance company to better understand all the different customer profiles.

With this information, it is expected that the company will be able to tailor the marketing approaches to different groups of customers, which will allow a significant reduction in expenses and, in addition, improve overall customer satisfaction by having targeted advertising to their needs.

We are expected to define clusters of clients and explaining our approach in order to solve the problem we have been assigned. Finally, we are also expected to explain the best marketing approach for each cluster defined.

2. Variable and Data Description

As stated in the project description sheet, this company's data is from 2016. In our notebook, we started by loading the SAS data into a Pandas DataFrame. After loading, we decided to check the number of observations in our Dataset and verified that there were 10296 records. We checked the variable types and made the **Table I, present in Appendix II.**

In a small analysis we tried to understand the behaviour of the *GeoLiveArea* variable, as the project sheet has no information on it, however without success. We also noticed some Premiums had a negative value, which can happen due to the cancelation of the insurance by a customer, therefore we assumed them as normal values.

3. Data Preparation

Initially, for ease of use, we converted the *CustID* column to an integer. This was done in order to be able to set it as the index, so instead of addressing to a customer as a position in a table, we could directly use the Customer ID.

After this, we checked for duplicated records. We found out we had 3 duplicated records and decided to drop them. Instead of having the starting 10296 observations, we now have 10293 observations.

In the next step we checked for null values in all the variables. We had 30 null values in *FirstPolYear*, 17 in *BirthYear*, 17 in *EducDeg*, 36 in *MonthSal*, 1 in *GeoLiveArea*, 21 in *Children*, 0 in *CustMonVal*, 0 in *ClaimsRate*, 34 in *PremMotor*, 0 in *PremHousehold*, 43 in *PremHealth*, 104 in *PremLife* and 86 in *PremWork*. At the end of this analysis, we decided to delete only the 17 values associated with the *EdugDeg* variable, so that it will be possible to process it right away. This variable was an object type variable, so we wanted to convert it to a format where we could analyze, interpret and process it more easily. Having nulls here would break the process. We will later deal with the remaining null values.

With this logic in mind, we decided to encode this variable. By observation, we noticed it was a String encoded into Byte form. We started by decoding it to String type, and we realized that all had a similar structure to "<NUMBER> - <Degree>". Taking this into account, we decided to manually encode it into a number.

3.1. Descriptive Statistics

After pre-processing the data, we applied some descriptive methods to obtain a table that describes all our variables, **Table 2, available in the Appendix II**. After a brief analysis, it raised some concerns about the coherence of the data. One example was the Customer Monetary Value having a minimum value a lot lower than the 25% percentile mark. The other anomaly was the birthyear of a customer being 1028. This customer was 993 years old.

3.2. Coherence Checking

In our coherence checking step, we decided to create new binary variables related to the coherence of each feature, for an easy analysis and later counting. These new variables have a value of 1 if they are inconsistent and 0 if not. We started by creating a variable that checks the BirthYear variable. This variable would report if a customer was born after 2021 or before 1903. Currently we are in the year 2021, so it is not possible to be born after this year. We also checked that the oldest person alive in the world is 118 years old, and was born in 1903, therefore we used this year. In our analysis, we realized only one record was not coherent here, so we decided to drop it.

The next step was creating a check for the first contract year for a particular customer. Initially, it didn't seem possible to us to have an insurance made before the person who created it was born. We also checked for insurances that were created in the future (after 2021) or before 1903, for a similar reason as the customer birth year. After our analysis, we realized there were a lot of observations here, specifically 1997 observations. Because of this, we decided to try to assign a logical explanation that would justify this high prevalence of inconsistent values in our Dataset. We chose not to drop them and theorized these were insurance transfers between family members after the death of a relative, and similar situations.

The next check we made was related to a customer age. We investigated and noticed that the minimum legal age in Portugal to make an insurance contract is 14 years old, although many insurance companies will choose not to insure a 14-year-old individual. We made this check, but no observations were present in the range.

The last check we made was related to the Education Degree of the customer. It does not seem very valid to us that an individual under the age of 18 should hold a college degree. We made this check, but there were also no observations present.At the end of our Coherence Checking, we had a total of 10275 observations left in our dataset.

3.3. Missing Value Treatment

As mentioned before in the beginning of the Data Preparation chapter, we had a few missing values detected. We started by replacing the null values in all the premiums by 0. This was done with

the assumption that the client was not insured in these categories. Dropping data here could make us lose valuable information on other premiums the client has made.

The next variable we analyzed was the *GeoLiveArea*. It only had one single null observation. Since there is not much information on this variable, and it was just a single value, we decided to drop it. Regarding the variables *FirstPolYear*, *BirthYear* and *Children*, and after a discussion about potential solutions for the null values, we came to the conclusion that we had no reasonable solution to deal with the missing values, therefore we decided to drop them too.

Lastly, to figure the Monthly Salary of a customer, we used the RandomForestRegressor Machine Learning algorithm, in order to accurately predict the customer's salary. This was done because we thought that these occurrences could arise from a possible desire for anonymity by the customers regarding this topic, but they are still valid observations.

At the end of the analysis, we no longer had missing values, and we had a total of 10233 observations in our Dataset.

3.4. Outlier Treatment

For outlier treatment, we went with a visual approach. This approach was very simple, as we plotted the boxplot of every numeric variable and simply marked the threshold from which we considered to be an anomaly and not an outlier **(Figure 1, in the Associated Figures Appendix)**, to remove the excessive ones. In the end, we removed 2 records from *MonthSal*, 6 from *CustMonVal*, 4 from *ClaimsRate*, 6 from *PremMotor*, 3 from *PremHousehold*, 2 from *PremHealth*, 1 from *PremLife* and 2 from *PremWork*, resulting in the end, in a total of 10212 records remaining in our dataset. We decided to apply this more conservative approach to the treatment of extreme values, to try to preserve as many observations as possible and thus not lose potentially relevant and differentiating information. We plotted the graph after our treatment, present in the **Figure 2, in the Associated Figures Appendix.**

3.5. Correlations

Initially, to check for correlations, we decided to make a correlations heatmap with spearman's method, of all the metric variables. Due to this, we temporarily dropped *EducDeg*, *GeoLiveArea* and *Children*, since they are not metric features.

After making the correlations heatmap, we observed that *ClaimsRate* and *CustMonVal* had a correlation of -1. This happens because when a customer makes a claim, the company gets more expenses from such customer, therefore the customer monetary value goes down. Eventually, we dropped the *ClaimsRate* variable, as the correlation in general with other variables was higher than *CustMonVal* with all the other variables.

The second correlation we detected was *BirthYear* with *MonthSal*, of -0.9. This means that a customer that was born earlier in time tends to get more money. The behaviour is a tendency of society, as older people have been in the workforce for longer, which usually means they have been promoted to higher job titles and make more money. Similarly, to what we did with the previous situation, we dropped *BirthYear*, as it was more correlated to other variables, when compared to *MonthSal*.

We think this step is of enormous importance since we want to prevent as much as possible the introduction of duplicated information into our algorithms and, therefore, prevent the infusion of noise. The final map can be observed in **Figure 3, in the Associated Figures Appendix.**

4. Data Preprocessing

In this step, we started preprocessing the data, in order to prepare it for the clustering algorithms. We performed some Feature Engineering, Normalizations, Encoding, Dimensionality Reduction and did some final checks for outliers and data visualization.

4.1. Feature Engineering

In order to attempt to get better explaining ability and differentiative power for easier partition of our data, we performed an initial step of Feature Engineering. To start, we changed the *FirstPolYear* variable to a year value, instead of the year it started. Like this we could directly measure how long a customer was with the company, instead of having to calculate it manually. This makes it easier for analysis, instead of comparing years.

Since the premium variables are representing a total of a customer per year, we decided to change the *MonthSal* and *CustMonVal* variables to represent a value of year, instead of month. This makes it easier for comparison between variables. With the application of these transformations, we decided to rename these variables to *FirstPotAge*, *YearSal* and *CustYearVal* respectively.

Lastly, we decided to create a new variable called *PremiumSalaryRate*. This variable calculates the rate between how much a customer spends on premiums and how much does the customer makes per year. By making this variable, we expected to ease our work, and that it would aid us in clustering later.

Finally, we checked for correlations again, as we altered some variables and introduced a new one. The changed variables seemed to be okay, however the new *PremiumSalaryRate* had a strong negative correlation with *YearSal*. Since it also presented direct high correlations with the premium variables, we chose to drop it.

4.2. Data Normalization

In order to ensure that all numeric variables are on a similar scale of representativeness, we decided to normalize the metric features of our data. We used a MinMaxScaler in order to perform this step. After scaling, we checked the dataframe to see if it appeared to be scaled correctly. We replaced the metric features of the original dataframe with the scaled ones.

4.3. Encoding

To perform the clustering of our data, we needed to encode the categorical data. For such, we used the One-hot encoding technique that was instructed to us in class. We decided to follow this approach rather than, for example, label and ordinal encoding, in order not to introduce any kind of order or relationship in cases where it was previously non-existent. We did this by selecting the non-metric features of our data, performed the encoding using the OneHotEncoder class from the SKLearn

library, and transformed our data. After our data was encoded, we reassigned the encoded variables to the dataframe, dropping the old non-metric features.

4.4. Dimensionality Reduction

To ease our cluster analysis, we performed a dimensionality reduction method known as Principal Component Analysis, or PCA. With this, we were aiming to reduce the number of total components present in our data, by linearly combining similar variables into few components. Furthermore, we also thought that this procedure could be useful later on by helping us visualize our data by reducing the dimensionality of the input space. We started by plotting the scree plot and the variance plot **(Figure 4, in the Associated Figures Appendix)**. By analysis of these plots, we reached the conclusion that the ideal number of components was 3.

In the next step, we performed PCA again, but this time with a fixed number of components. After this analysis was done, we merged the new PCA variables into the main dataframe.

Still in this chapter, and for ease of use later, we decided to split the features into groups for easier addressing, separating metric, non-metric and principal components.

4.5. Outlier Analysis

In this step, we decided to perform some outlier analysis using DBScan. In the practical lessons, we noticed that DBScan seemed to be a very coarse clustering algorithm, that would generally separate the data into one giant cluster, and in noise, therefore we decided to experiment here with it, in order to identify possible outliers that we may have previously ignored.

To begin with, we had to adjust the hyper parameters of this clustering algorithm, namely the Epsilon. We did such by using the Nearest Neighbors search algorithm, calculate the distances and plot them. In this step, by visually analyzing the graph **(Figure 5, in the Associated Figures Appendix)**, we obtained an epsilon of 0.16. With this information, we ran the algorithm with an epsilon value of 0.16 and a min_samples of 16. This is an arbitrary value usually calculated by multiplying the number of features by 2.

After the execution of the algorithm, we were left with 2 clusters. A normal cluster with 9883 observations, and an outlier cluster with 329 observations. Here we separated the outliers from the normal dataset, for later assignment. At this point, our dataframe had 9883 observations.

4.6. Visualization of the Input Space

To visualize the distribution of our features and try to understand better their behavior, we used self-organizing maps. These maps distribute the observations in a topological structure, for ease of observation.

We started by visualizing the different component planes **(Figure 6, in the Associated Figures Appendix)** for each numeric feature, to visualize possible patterns. These Component planes gives us useful information regarding feature importance and the variability associated with each one of them. In addition to this, we can also try to observe correlations between variables and possible outliers that

we may have missed. With the component planes established, we realized that, except for *PremMotor*, all the premium related variables behaved in similar ways and, therefore, we can clearly say that these variables are spatial correlated. *CustYearVal* and *FirstPolAge* behaved inversely.

After this brief analysis, we plotted the UMatrix **(Figure 7, in the Associated Figures Appendix)** to check for potential clusters and outliers. With the UMatrix, we noticed a lot of units in the right lower section, which were very far away from the other units, well distributed along the matrix. This indicates potential outliers in that area, and a lower probability of forming a cluster there. Coincidently, this matrix matches with the average feature distribution of *PremLife*, *PremWork* and *PremHousehold*.

Finally, we decided to plot the Hitmap **(Figure 8, in the Associated Figures Appendix)** to get an idea of the number of observations allocated to each neuron in the SOM network. From this visualization we can infer the possibility of the existence of a cluster far from the others in the lower right zone of the network due to the high number of observation present. This may corroborate the idea we presented in the previous paragraph where we initially identified these occurences as possible outliers.

5. Clustering

Before starting to cluster our data, we initially split the variables into perspectives. We made a total of three perspectives, one for the Premiums, with all the Premium related variables, one for the Demographic and Socioeconomic features of the customers, that included the Yearly Salary, the education level, the unknown *GeoLivArea* variable and if the customer has children or not, and lastly one for the Contract and Company information on the customer, that contained the *CustYearVal* variable and the *FirstPolAge*, which contains the value in years of how long ago a customer made their first insurance Policy. With this approach, we had in mind to apply several clustering algorithms to each perspective created and, through the obtained R2 and silhouette scores, apply those that seemed more efficient to later combine them in a final clustering solution.

5.1. Clustering the Premium Perspective

In our project, we decided to start by clustering the Premium Perspective we mentioned previously, also taking into account that all algorithms were applied with the assumption that the variables present in this perspective are all numerical. After reviewing and remembering some advice from the theoretical classes, we decided to initially apply a KMeans algorithm, with an arbitrary high number of clusters, in order to obtain a smaller set of data representative enough of the original observations. Since we strongly believe that the hierarchical clustering algorithm is a suboptimal algorithm when compared to the KMeans, we decided that it would make more sense to get the optimal number of clusters through the last. We obtained 20 centroids, and then applied hierarchical clustering on these centroids, and plotted Ward's Dendrogram for our clustering **(Figure 9, in the Associated Figures Appendix)**. By observation, we realized our data had a total of 4 clusters.

After obtaining the ideal number of clusters for our data, we performed another KMeans clustering, this time with the ideal number of clusters, and plotted the Silhouette plot **(Figure 10, in the**

Associated Figures Appendix) for these same clusters, obtaining a result of 0.3513. Lastly, we obtained the R2 Score for our execution of the KMeans algorithm, which was 0.7272.

The next step was to perform Hierarchical clustering on our data, with the ideal number of clusters, 4, set. In tests before, we noticed that performing hierarchical clustering using Ward's method always yielded better results than with other methods, so our choice here was to use only Ward's method, to save execution time. Similarly, to our previous KMeans execution, we plotted the Silhouette plot **(Figure 11, in the Associated Figures Appendix)** for the various clusters, to understand their distribution, giving 0.3116 as the final score. Lastly, we obtained the R2 Score for our execution of the Hierarchical clustering algorithm, which was 0.6801.

After this, we performed the DBScan algorithm. In previous tests, it always identified only 1 cluster with noise, so we had doubts it would be as accurate as other algorithms, especially because we cannot insert a custom number of clusters. We started by using the KNearestNeighbors algorithm so that we could plot the K-Distance Graph. With this graph, we could obtain the Epsilon parameter for our algorithm. After analysis of this graph, we obtained an epsilon value of 0.10. The next parameter that DBScan requires is min_samples, which can be arbitrary by getting the number of features of the data to be analyzed and multiplying by 2. For this Perspective, it was a total of 10 min_samples. After running the algorithm, we did not plot the silhouette plot, as the value of the R2 score was just 0.0897, and that the algorithm identified only one cluster.

Lastly, we used Self Organizing Maps (SOM), with Hierarchical Clustering. The parameters used for hierarchical clustering were like what we mentioned previously, 4 clusters and using Ward's method. We obtained the map **(Figure 12, in the Associated Figures Appendix)** of the cluster distribution and calculated the R2. With this algorithm, we obtained 0.6897 of score.

Finally, since the KMeans Algorithm yielded the best scores, we clustered this perspective with it, obtaining the cluster distribution map **(Figure 13, in the Associated Figures Appendix)**.

5.2. Clustering the Demographic Perspective

The demographic perspective, as mentioned earlier, included a lot of non-metric features, therefore we had to use the KPrototypes algorithm instead of the KMeans algorithm, which can handle mixed data types and we also did not use DBScan or SOM here since they cannot handle categorical data very well. In a similar fashion to the previous iteration, we started by determining the ideal number of clusters, by setting the number of clusters of the KPrototypes algorithm to an arbitrary higher value of clusters (20). Following similar setups to what was shown in the practical lessons, we used Huang's initialization and a total of 15 initializations. We also set a random state seed, to be able to replicate these results. We obtained a total of 20 centroids, as expected. In these centroids, we applied Hierarchical Clustering, and plotted Ward's Dendrogram. By observation we noticed that the ideal number of clusters for this Perspective was 3 **(Figure 14, in the Associated Figures Appendix)**.

The next step was running the KPrototypes algorithm, this time with the correct number of clusters. We plotted the silhouette plot **(Figure 15, in the Associated Figures Appendix)**, similarly to

what we did in the previous perspective, and calculated the R2 Scores for the demographic variables. For KPrototypes, the final score was 0.2331 and 0.4329 respectively.

After this, we performed Hierarchical clustering, in a similar fashion, with the correct number of clusters set. We plotted the silhouette plot **(Figure 16, in the Associated Figures Appendix)**, which appeared to look better, compared to the KPrototypes resulting plot, giving a score of 0.3024. The final R2 Score was not still ideal, but was better when compared to KPrototypes, with a value of 0.4552.

Finally, since the Hierarchical clustering yielded better results, we used it to obtain the final clusters for this perspective.

5.3. Clustering the Contract Perspective

The contract perspective was composed of leftover information about the customer contract. This information was metric, therefore the plan of action was equal to the Premium perspective. We started by using a KMeans algorithm, with 20 clusters set, as explained. After obtaining the centroids, we used Hierarchical clustering on these centroids, and plotted Ward's dendrogram. By observation, we realized the ideal number of clusters was 4 **(Figure 17, in the Associated Figures Appendix)**.

The next step was doing a KMeans algorithm, this time with the ideal number of clusters set. After performing the algorithm, we also plotted the silhouette plot **(Figure 18, in the Associated Figures Appendix)**, which appeared to be the best of all the plots we made, having nearly no observations in the wrong clusters and giving a final score of 0.5964. After this, we calculated the R2 Score, which was 0.8927, our best.

After doing the KMeans algorithm, and following a similar plan to the Premium perspective, we performed Hierarchical clustering, with a set number of clusters. As mentioned before, we used Ward's method. After clustering, we plotted the silhouette plot **(Figure 19, in the Associated Figures Appendix)**, which was not as good as the previous plot, having some data attributed to the wrong clusters with a score of 0.5715. We also calculated the R2 score, which was 0.8805.

The next step was performing the DBScan algorithm, in a similar way of how we did with the Premium perspective. Unfortunately, the results in this perspective were as bad as in the premiums, having only 1 cluster and noise. The R2 score obtained was of 0.0132.

Lastly, we performed SOM, with Hierarchical clustering. We used the ideal number of clusters, 4, and Ward's linkage method. The hittmap obtained **(Figure 20, in the Associated Figures Appendix)** of the cluster distribution looked good and appeared that all the records were well clustered. Finally, we calculated the R2 Score, which was 0.8335.

Finally, having yielded the best results with the KMeans algorithm, it was selected as our final algorithm for this perspective, and we plotted the cluster map **(Figure 21, in the Associated Figures Appendix).**

5.4. Merging the Clusters

As mentioned before, for the Premium and Contract Perspective, the best results were yielded with the KMeans algorithm, while for the Demographic perspective, the best results were yielded with Hierarchical Clustering, using Ward's method. To start, we made a pivot table to better observe our data now allocated to the 60 clusters coming from the 3 different perspectives. We calculated the mean and median, in order to try to define a threshold that would allow us to group smaller clusters with larger ones. Eventually, we decided to merge using Hierarchical Clustering, as it seemed more efficient.

To merge with Hierarchical Clustering, we started by obtaining the cluster centroids for all the concatenated clusters from the different perspectives. We did not set the ideal number of clusters this time, to reduce bias. We used Ward's method for Linkage, and a distance threshold of 0 in order to plot the full grown Dendrogram **(Figure 22, in the Associated Figures Appendix)**. After performing hierarchical clustering, we made Ward's Dendrogram for the data, for better observation. By visual observation, we noticed there were 4 clusters present.

With this information, we re-ran the algorithm, this time with the correct number of clusters set. We then mapped the hierarchical clusters and concatenated clusters, so that we would merge the centroids together. After this, we made the pivot table of the new created clusters and distributions. We obtained 4 clusters as expected, the first with 2345 observations, the second with 2711 observations, the third with 1415 observations and the fourth with 3412 observations.

6. Cluster Analysis

With the clusters created all we needed to do now was perform a small analysis on them, to understand their behavior and obtain conclusions. Initially we started by evaluate the proportions of the categorical variables in the different clusters, moving then to profiling the clusters and assessing the feature importance. In a final step, we reclassified outliers.

6.1. Interpreting categorical variables

Firstly, we decided to evaluate the proportions of the different encoded categorical variables in the clusters that resulted from the merge we just performed. Here, we check the averages for each feature corresponding to each cluster and draw some potentially relevant conclusions: 88% of the individuals in cluster 3 have children; approximately 74% of individuals in cluster 3 have high school or Bsc/Msc degree, which makes this cluster the one with the highest level of education.

6.2. Cluster Profiling

In this step, we profiled the clusters, separating them by perspectives, and in the end, we presented a graph of the merged results **(Figure 23, in the Associated Figures Appendix)**. In this chapter, we will not detail our explanation of the plotted graph, as it is a topic that will make the next chapter irrelevant. After plotting the profiling graphs, we plotted the T-Distributed Stochastic Neighbor Embedding (TSNE) graph **(Figure 24, in the Associated Figures Appendix)**, to visualize our data in a 2-dimensional map, separated by clusters.

6.3. Assessing the feature importance and reclassifying outliers

To assess the feature importance of our data, we decomposed the R2 Score for each cluster into the R2 Score for each variable, using a set of functions that we learned in the practical classes. We performed this analysis for the metric features.

In this step, the Premium related variables had a lot more importance than the customer and contract data variables. The highest rated variable was PremMotor, with 86% of R2 Score. The Premium variable with the lowest rating was PremWork, with 43% of R2 Score. Looking at the customer and contract data variables, the highest rated variable was YearSal, with just 8% of R2 Score. The lowest rate variable was FirstPolAge with 0.04% of R2 Score.

Lastly, we reclassified the Outliers. In this step we used a decision tree. Before using a decision tree, we experimented with other Classifier algorithms, however they did not yield a good result. Decision Trees were able to predict the cluster that the outliers belonged to with 90.64% of accuracy.

7. Marketing Approach

As mentioned before, we separated our customers in 4 clusters. Each cluster presents a different behavior according to the profiling graph. Therefore, an individual analysis is needed.

The customers in the Cluster 0 tend to be median customers in *PremMotor*, *PremHousehold*, *PremLife* and *PremWork*. Therefore, customers display expansion in these areas, and the advertisements should be targeted in this way. These customers are the highest in *PremHealth*, so there is no need in advertising in this area. Like so, we labeled these customers “Growing-Health”. The label comes from the fact they are growing in all areas, but already have a lot of Health insurances.

The customers in cluster 1 tend to be median customers in all the Premium variables. We believe that in this area, the advertisement should be even to all the services the company presents, as they have potential of growing. We labeled these customers “Growing”, as they are growing in every premium variable.

The customers in cluster 2 are the lowest in *PremMotor* and the highest in *PremHousehold*, *PremLife* and *PremWork*. They are a median in *PremHealth*, therefore this cluster seems to be in expansion in the Health Insurance area, and the company should invest in advertisements related to Health for this cluster. We labeled these customers “CityResidents”, as they likely do not own a motor vehicle and could rely on public transportation.

Lastly, the customers in cluster 3 are the lowest in all areas except *PremMotor*. These customers only have automotive insurance in the company. Although being at a low point in *PremHealth*, it appears that it is an area in expansion, so we recommend targeting advertisements to that area, but focusing on other clusters mainly. We labeled these customers “Drivers”, as they only have vehicle related insurances.

Appendix I – Associated Figures

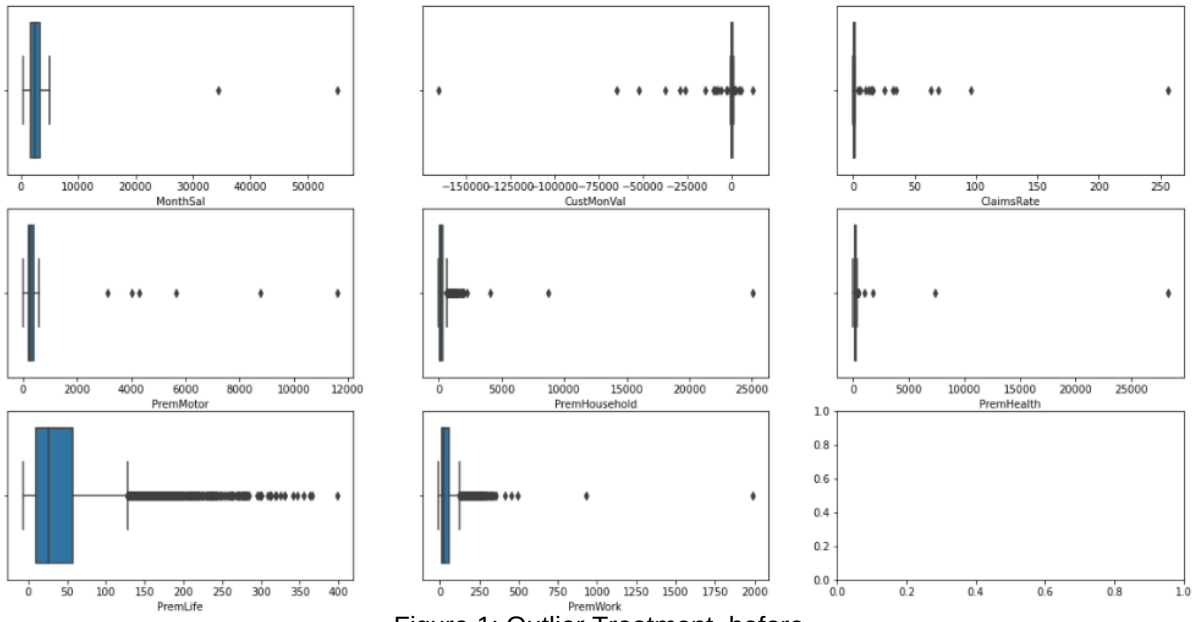

Figure 1: Outlier Treatment, before

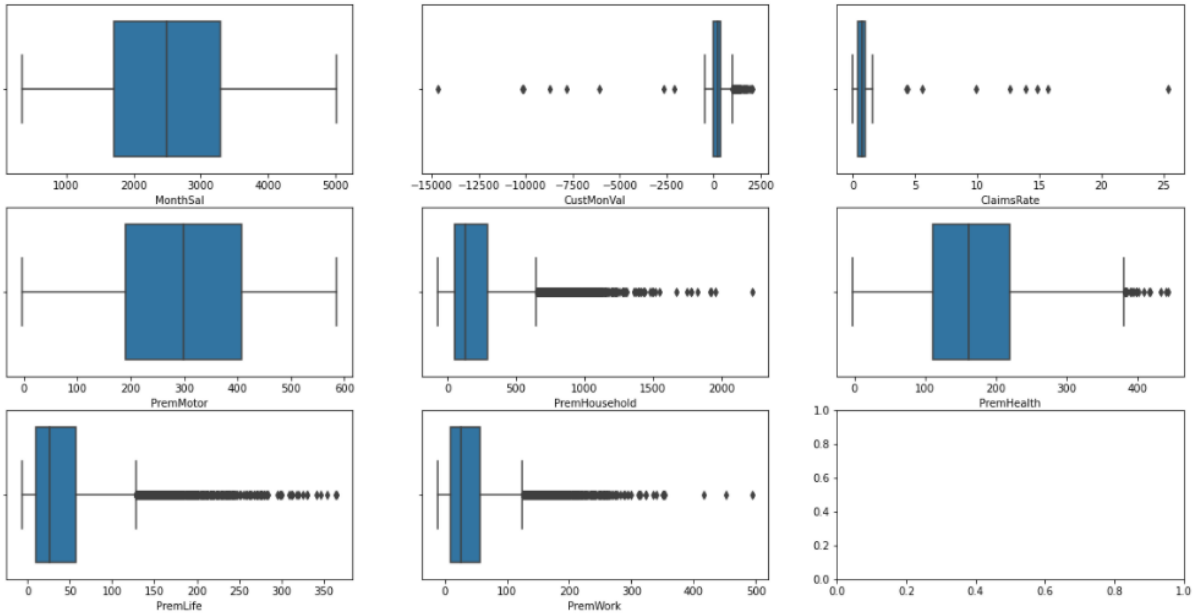

Figure 2: Outlier Treatment, aft

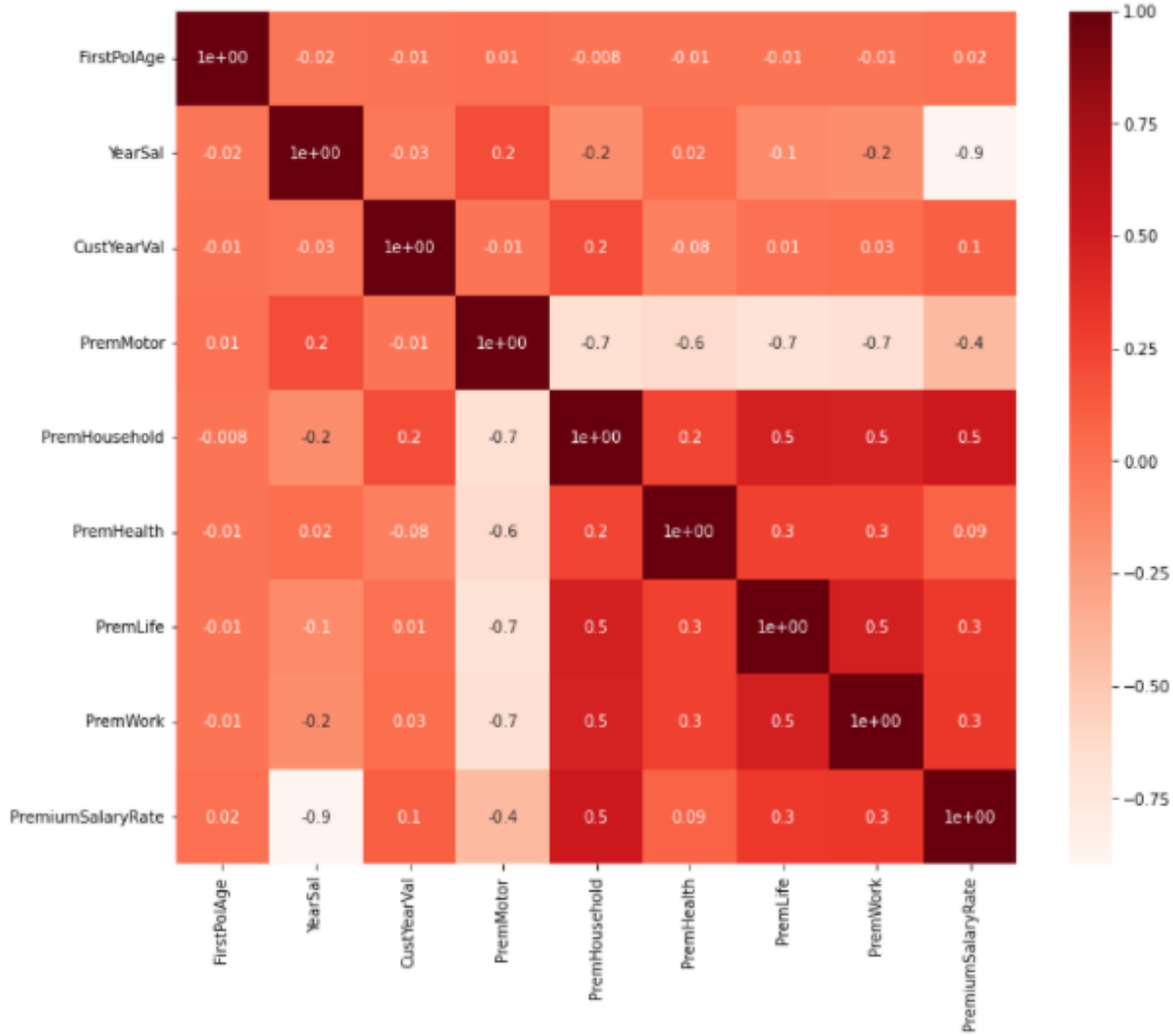

Figure 3: Correlations heatmap

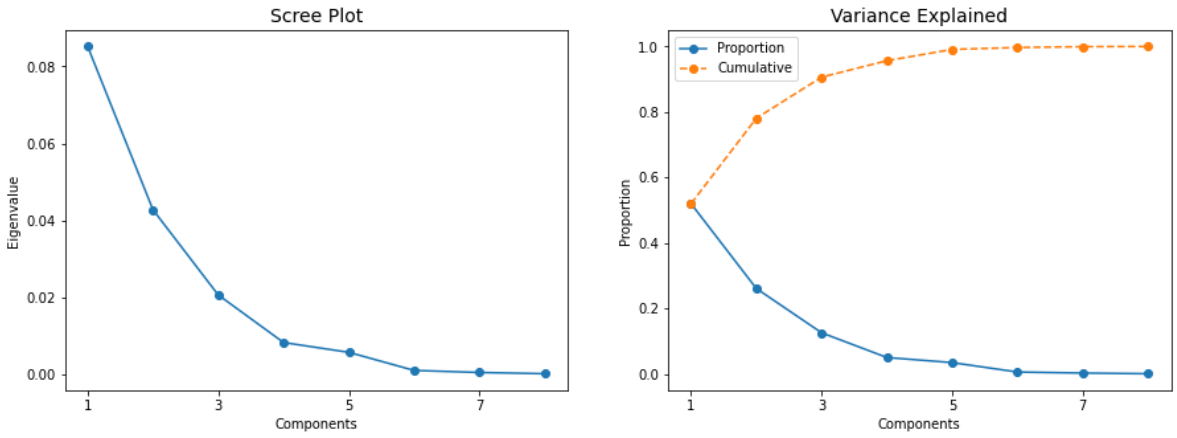

Figure 4: PCA diagrams, for dimensionality reduction

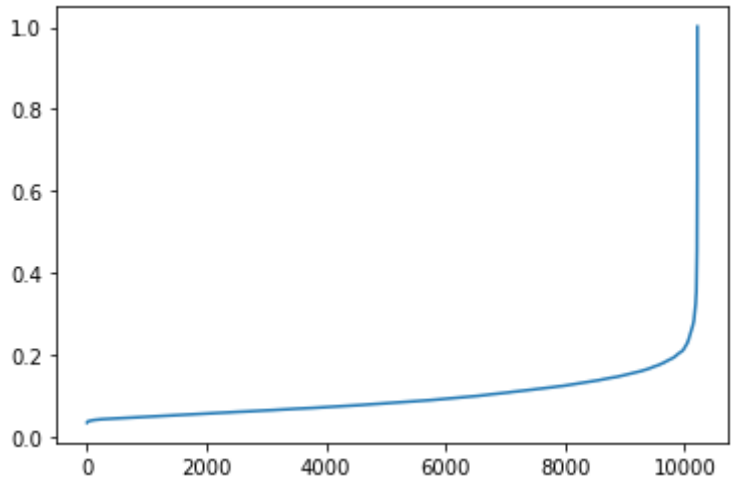

Figure 5: Nearest Neighbors plot for DBScan's Epsilon parameter

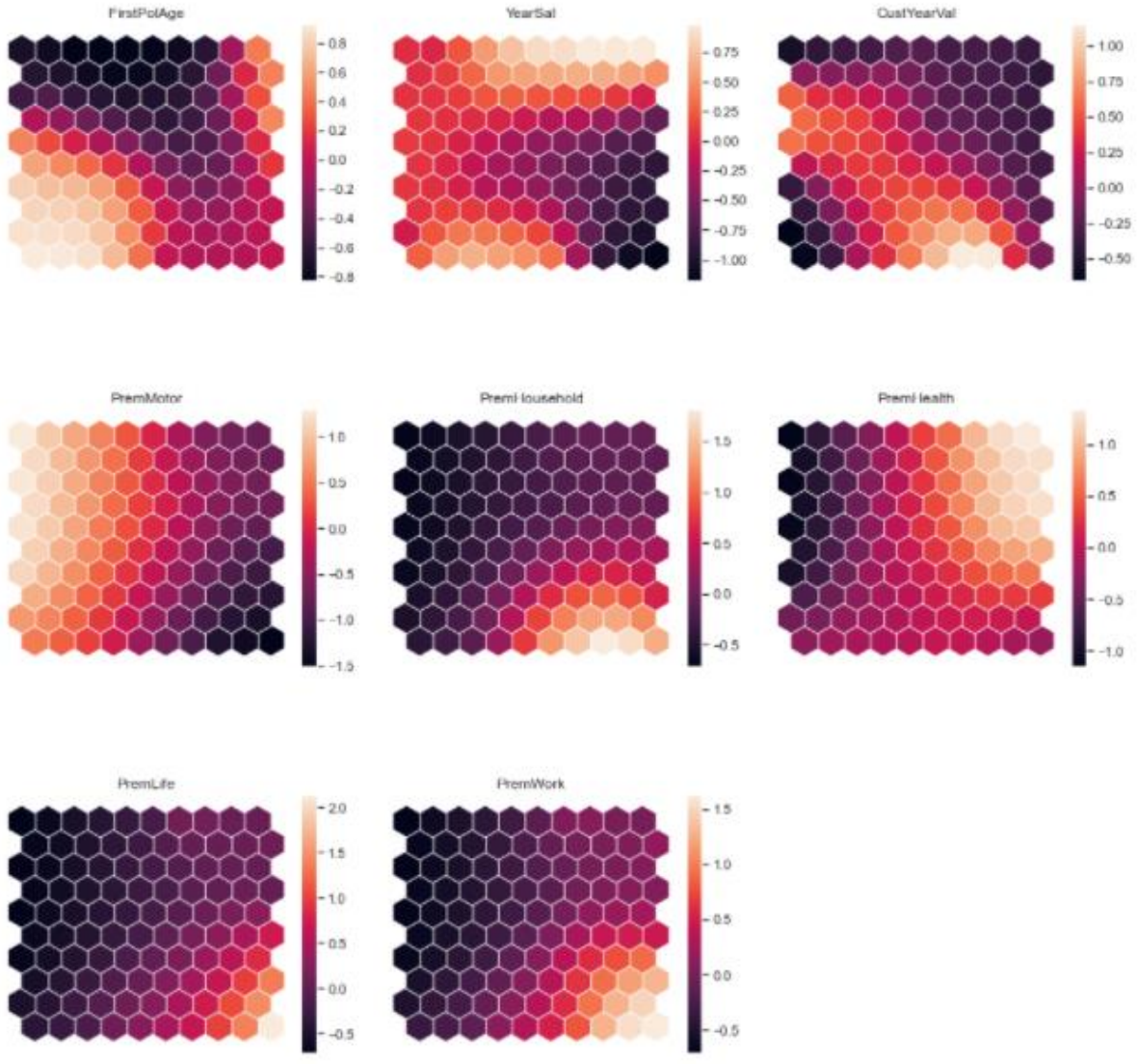

Figure 6: Component planes

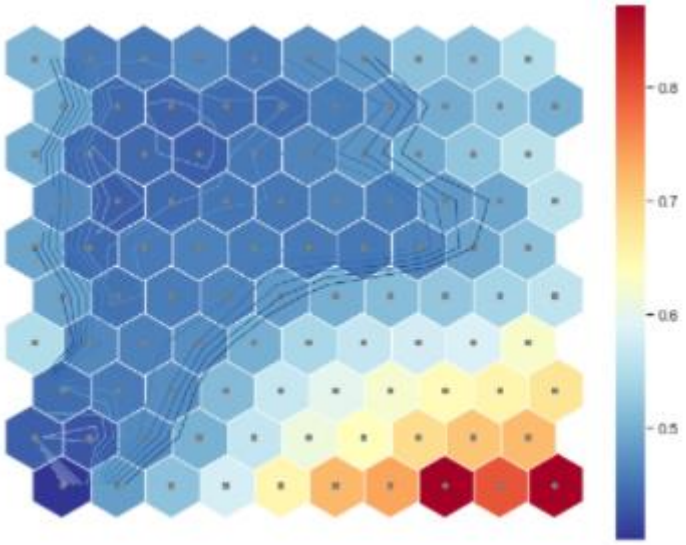

Figure 7: UMatrix

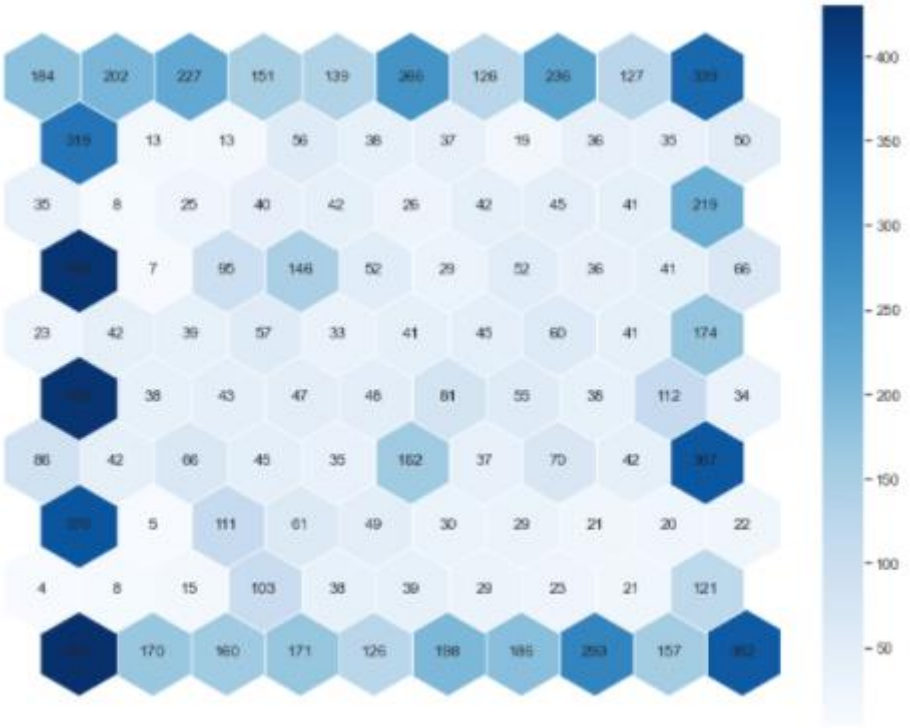

Figure 8: Hit Map

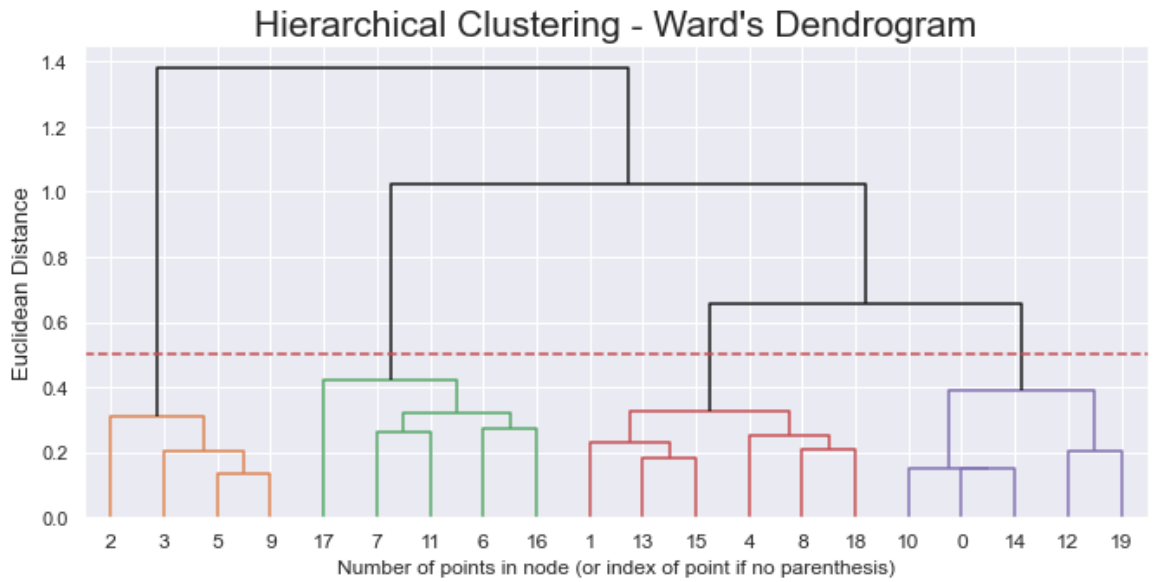

Figure 9: Ward's Dendrogram for determining the number of clusters in the Premium Perspectives

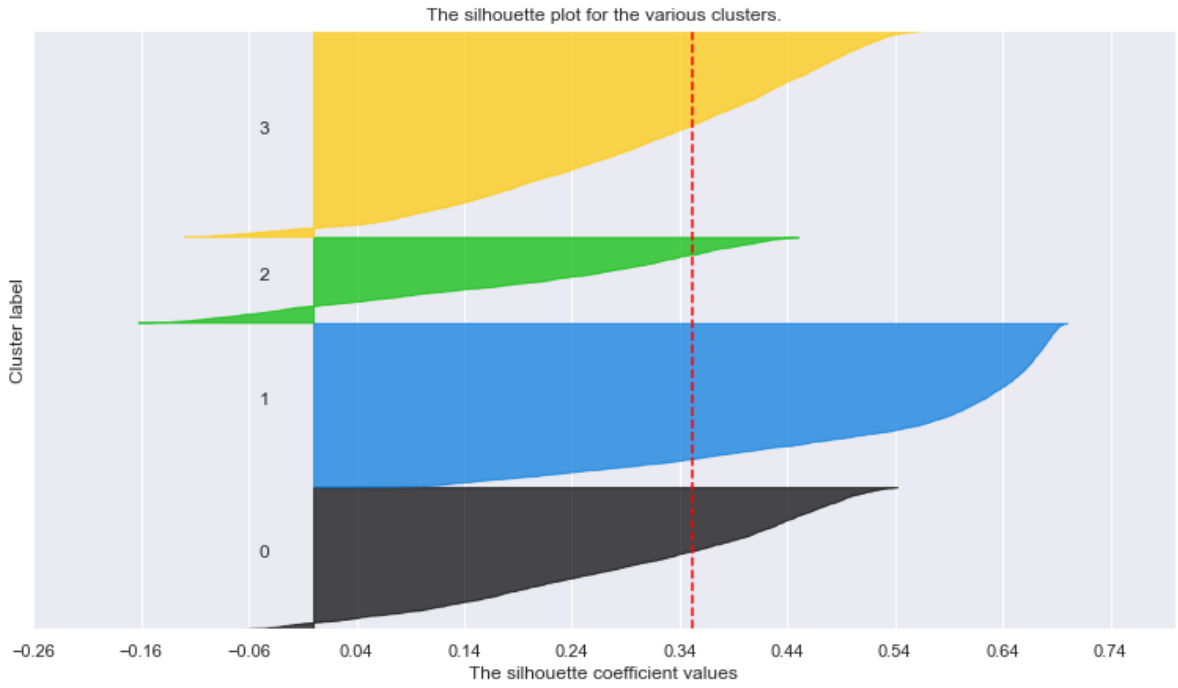

Figure 10: K-Means Silhouette Plot for the Premium Perspective

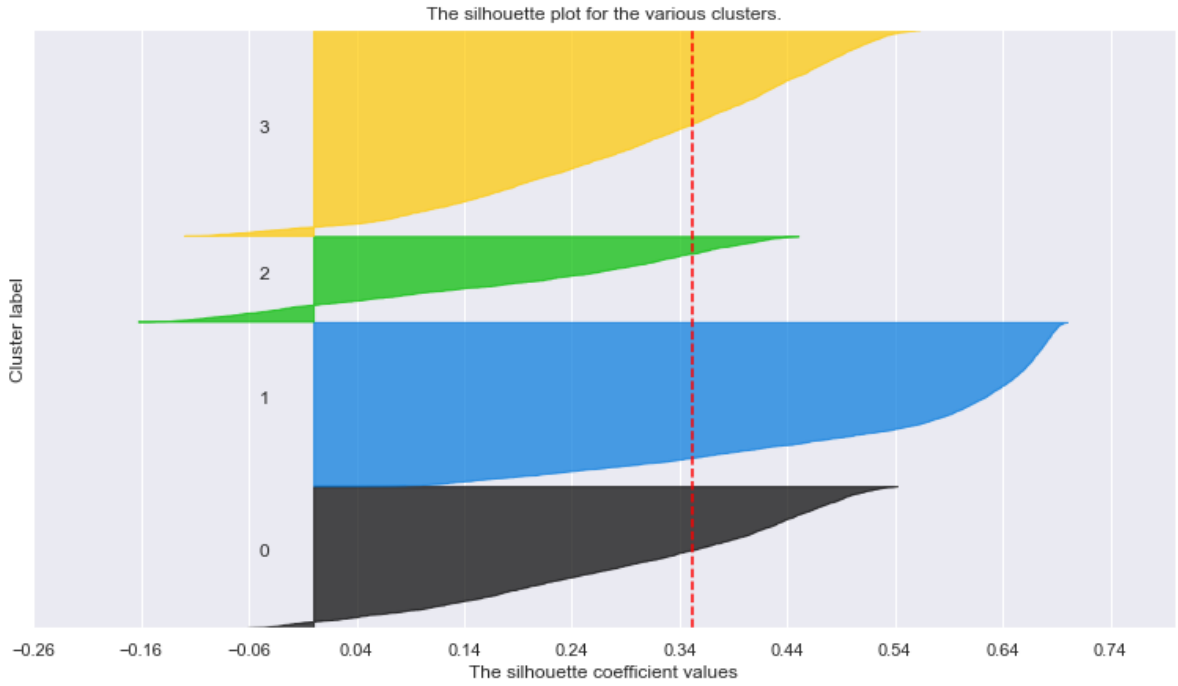

Figure 11: Hierarchical clustering silhouette plot for the Premium Perspective

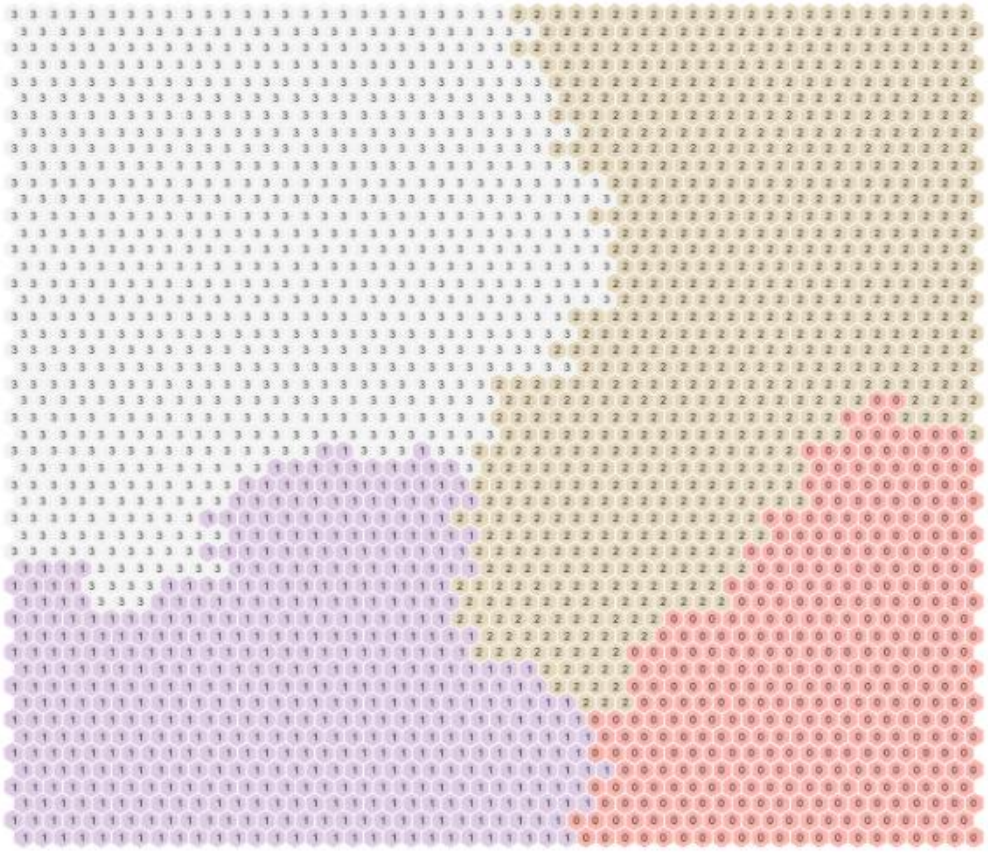

Figure 12: Map of clusters of the Premium Perspective

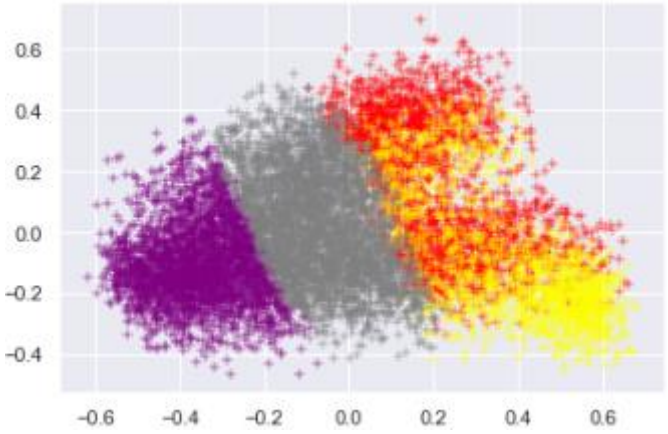

Figure 13: Final cluster map for the Premiums perspective

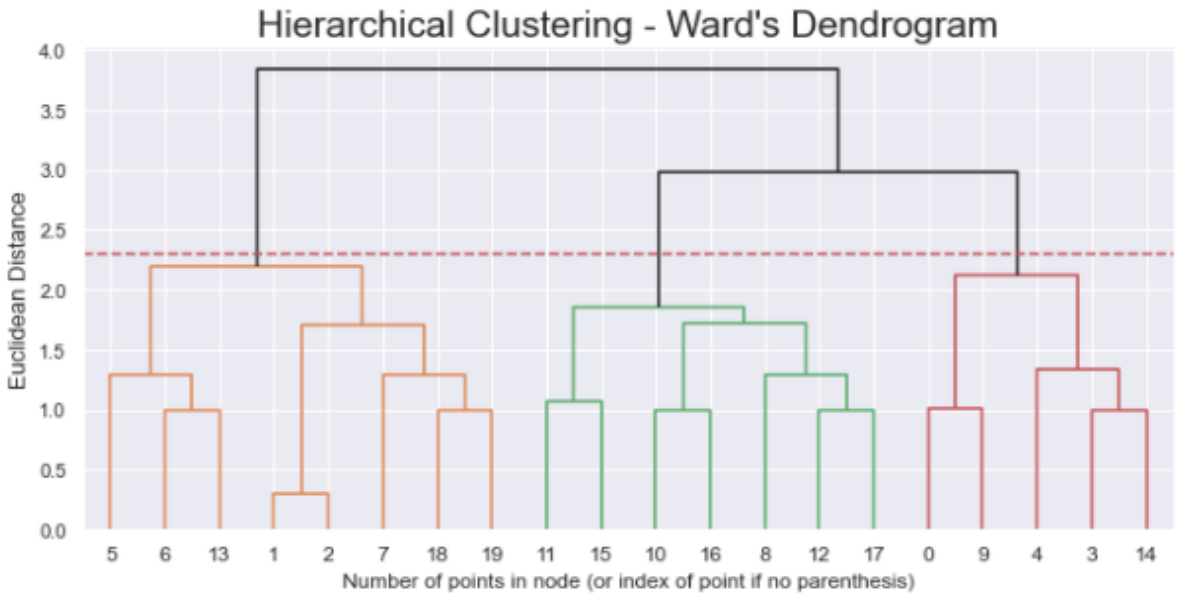

Figure 14: Ward's Dendrogram for determining the number of clusters in the Demographic Perspectives

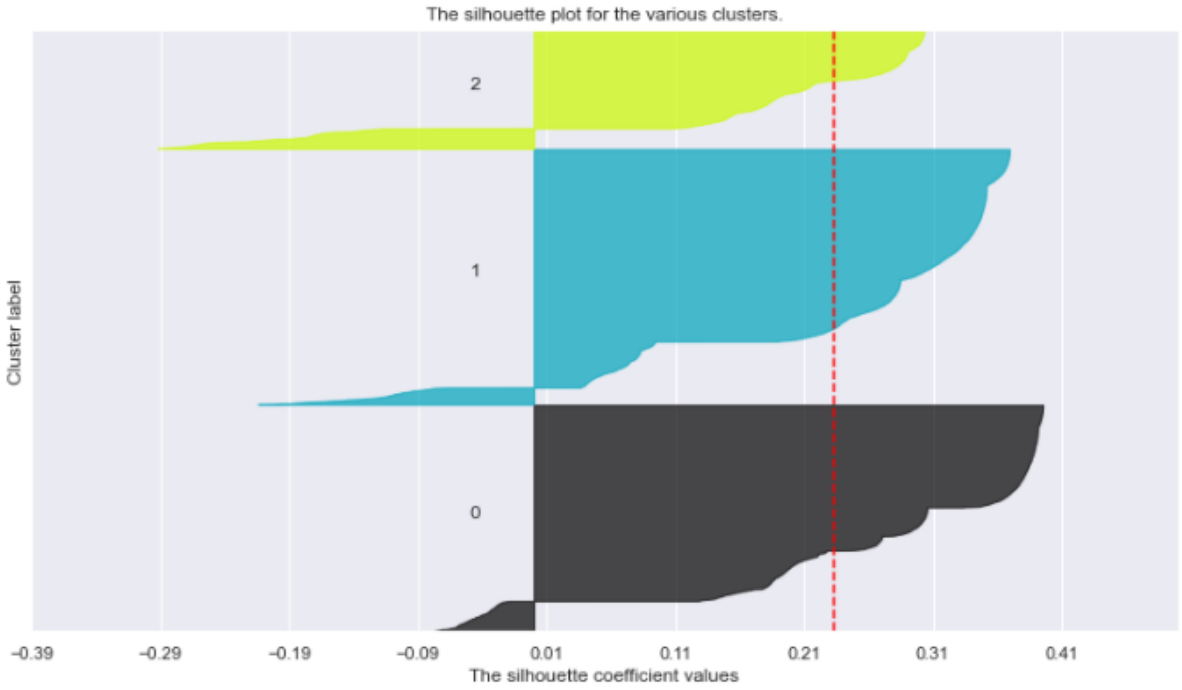

Figure 15: K-Prototypes Silhouette Plot for the Demographic Perspective

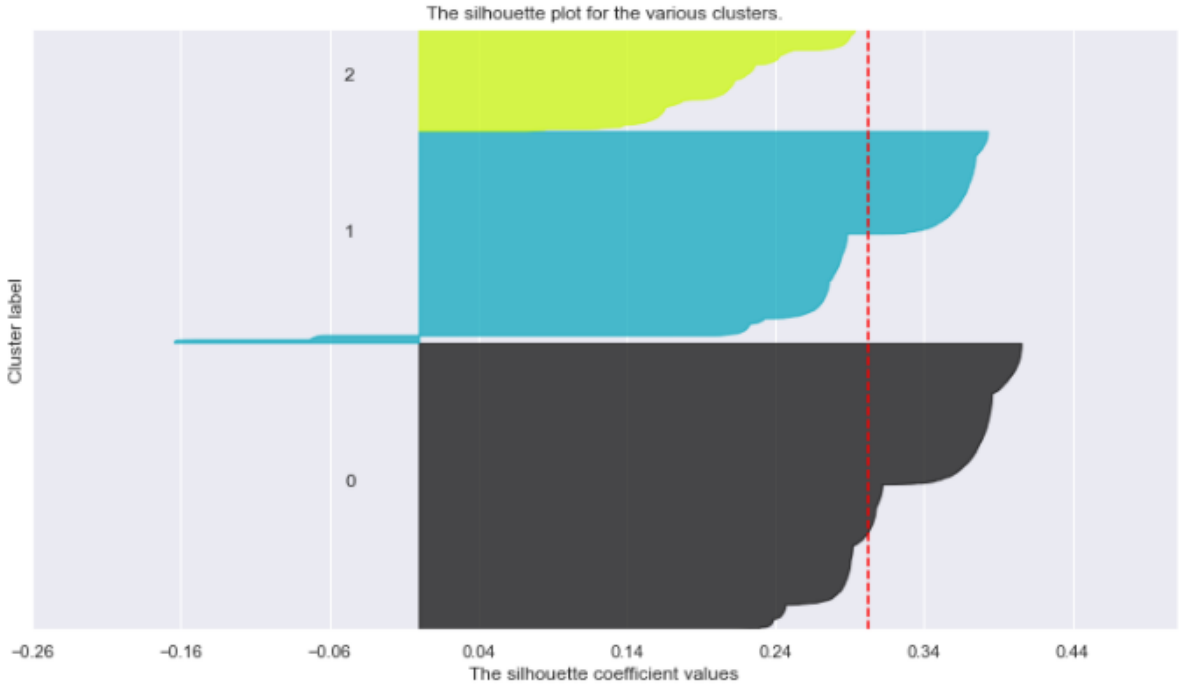

Figure 16: Hierarchical Silhouette Plot for the Demographic Perspective

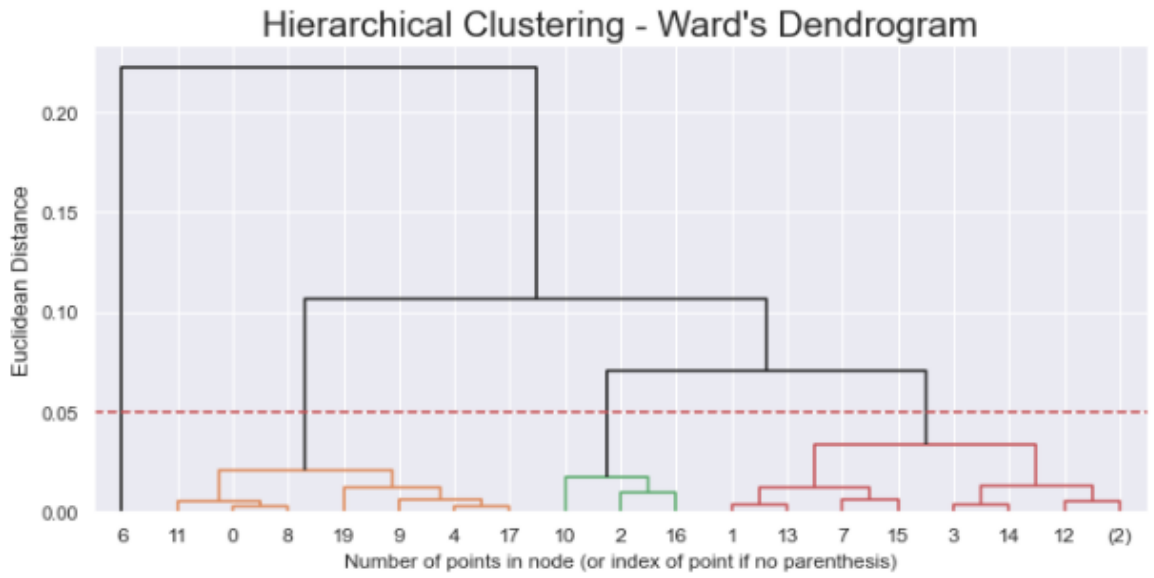

Figure 17: Ward's Dendrogram for determining the number of clusters in the Contract Perspectives

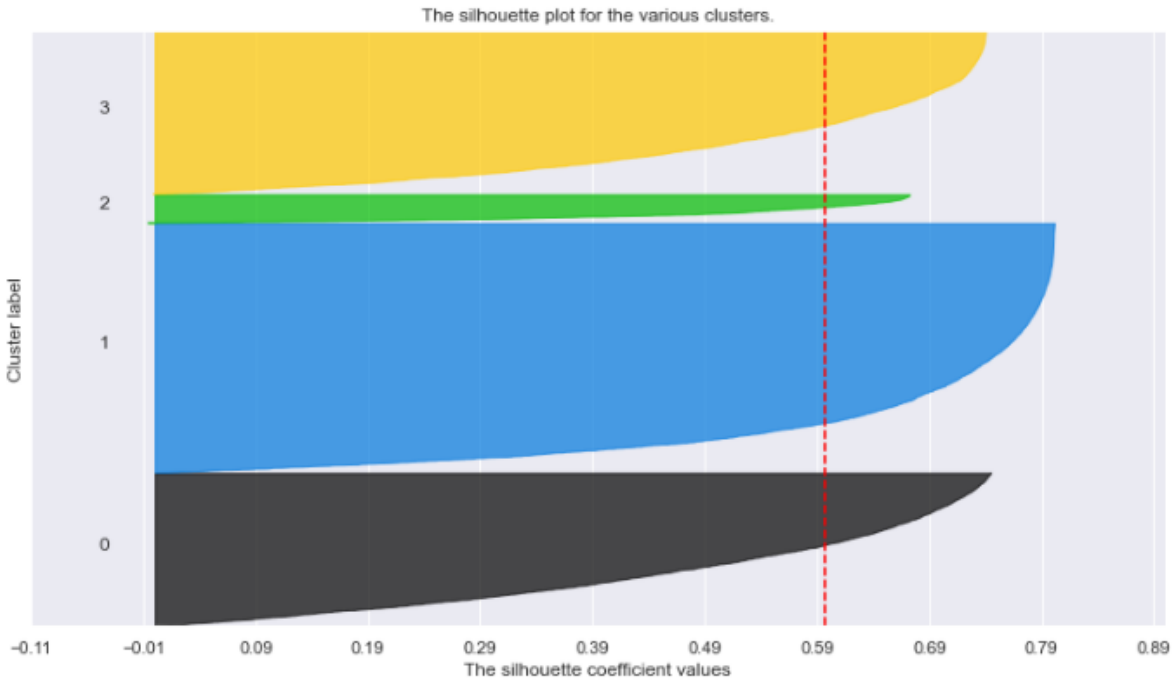

Figure 18: K-Means Silhouette Plot for the Contract Perspective

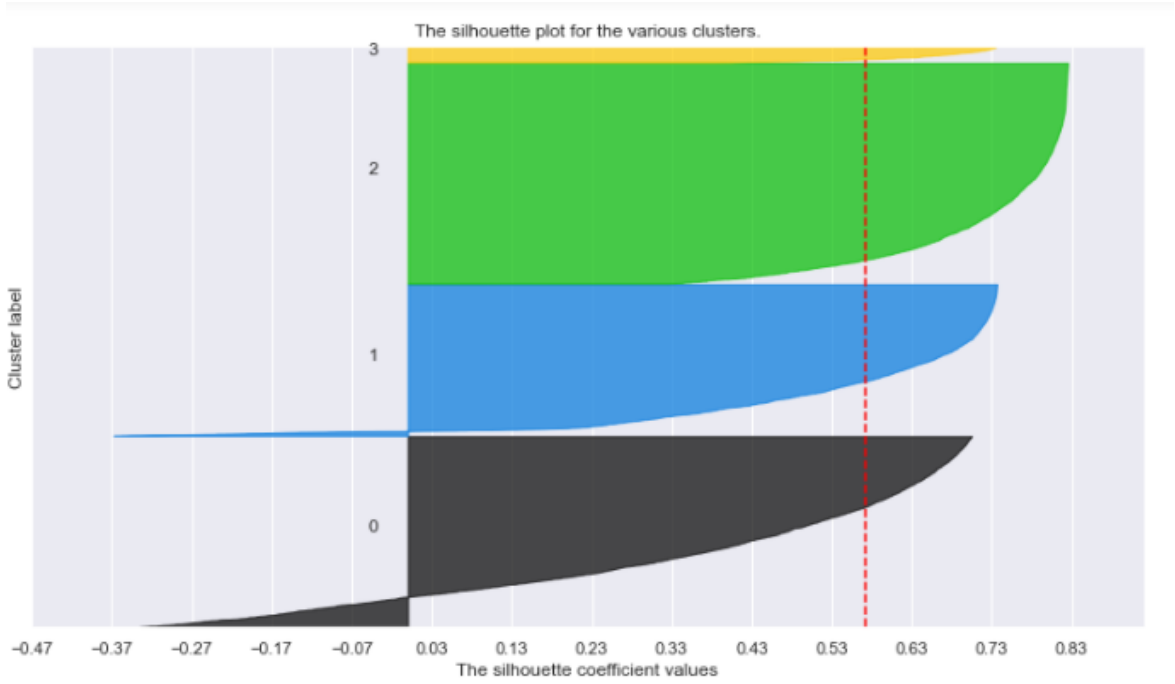

Figure 19: Hierarchical clustering silhouette plot for the Contract Perspective

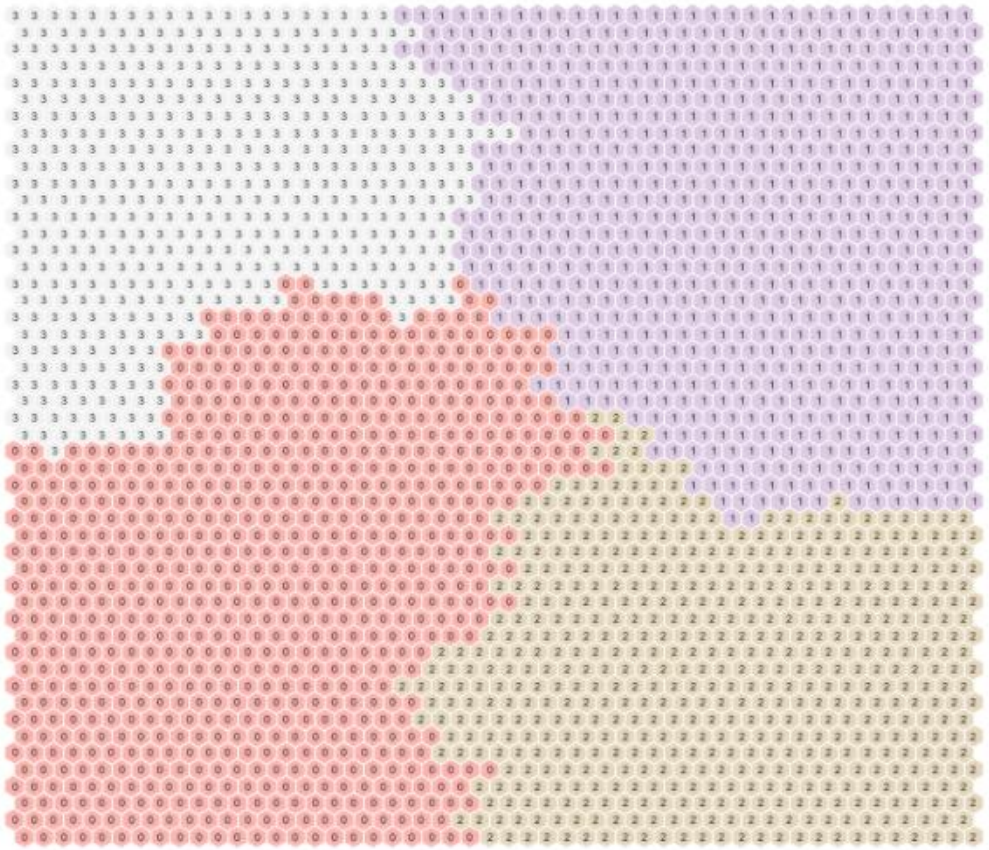

Figure 20: Map of clusters of the Contract Perspective

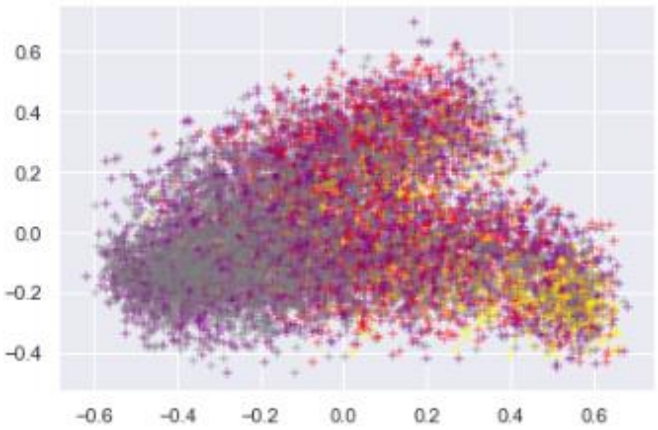

Figure 21: Final cluster map for the Contract perspective

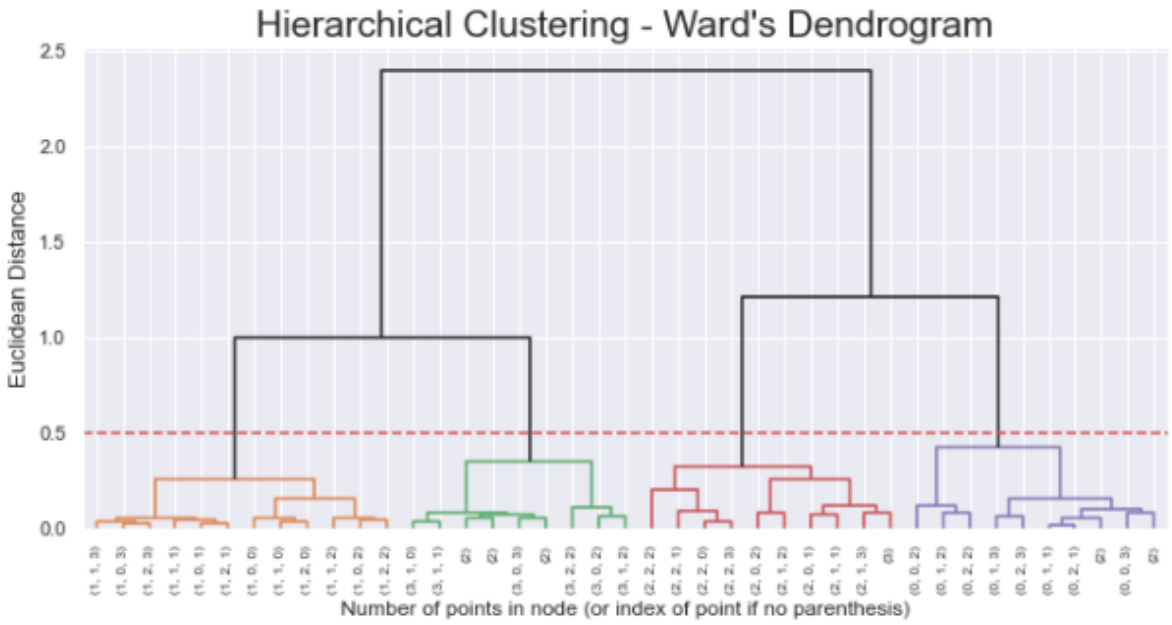

Figure 22: Dendrogram for the cluster merging

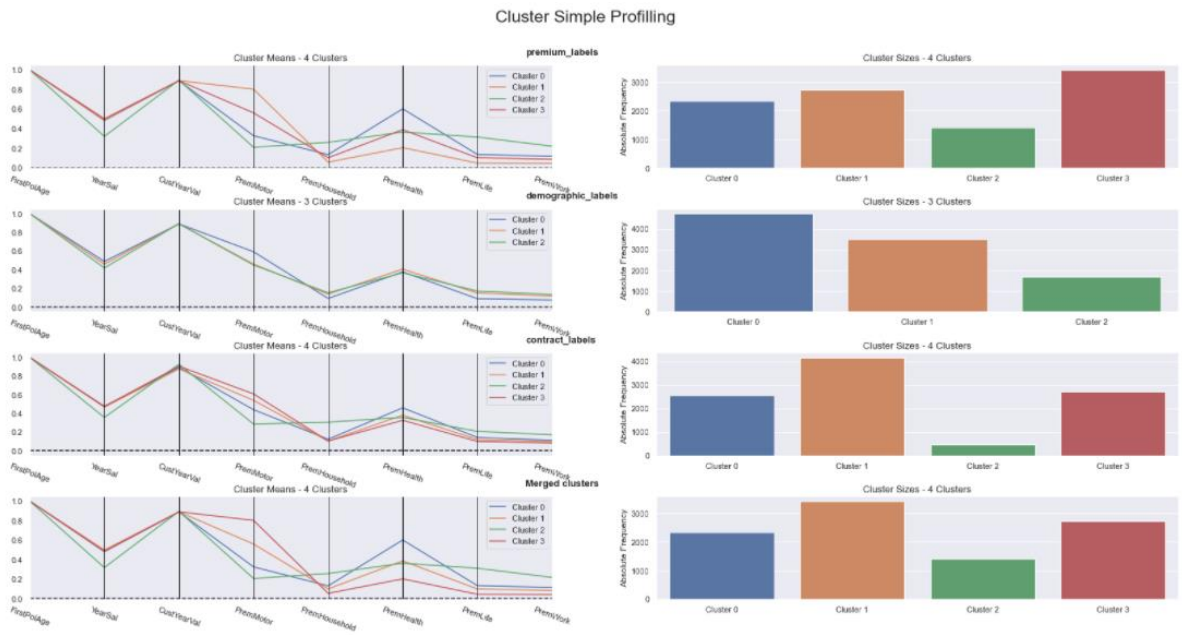

Figure 23: Cluster Profiling graph

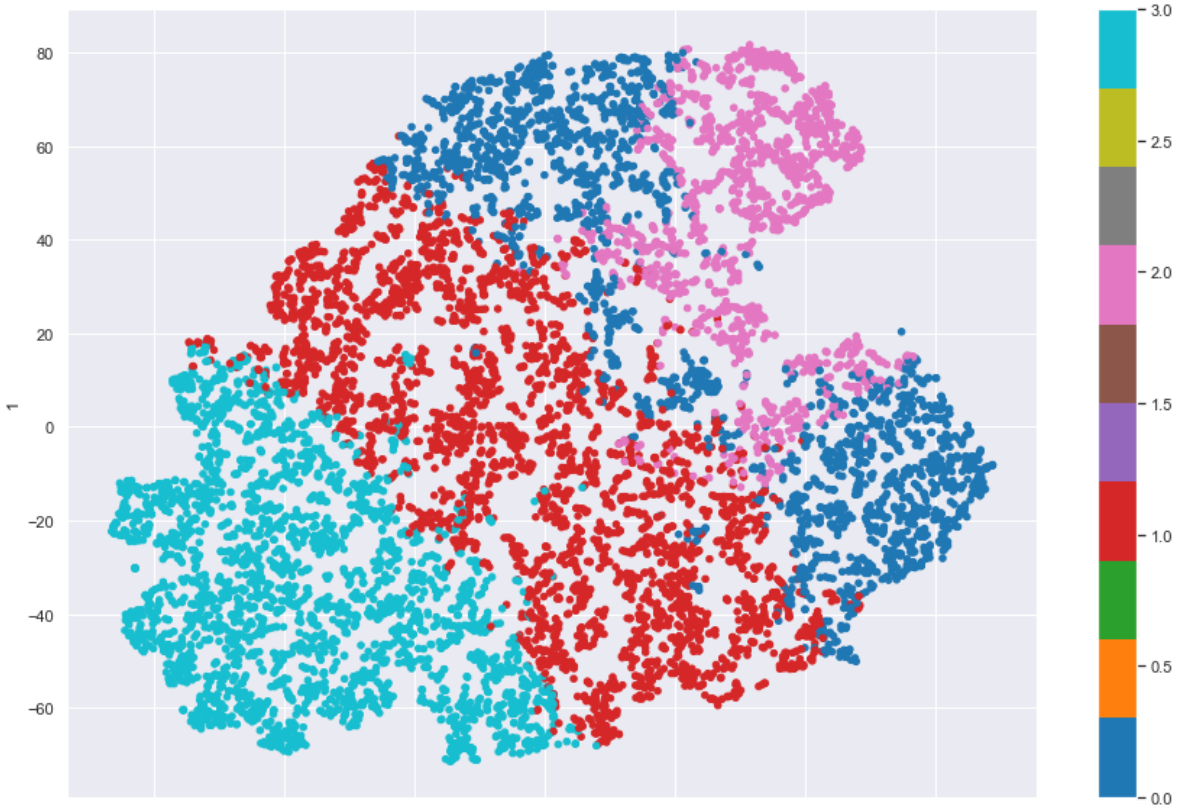

Figure 24: t-SNE graph

Appendix II – Tables

Variable Name	Datatype	Description
CustID	float64	Customer ID
FirstPolYear	float64	First Year as a Customer
BirthYear	float64	Customer's Birthday Year
EducDeg	object	Customer's Academic Degree
MonthSal	float64	Gross Monthly Salary
GeoLivArea	float64	Living Area (no more information on this)
Children	float64	Binary Variable of Children
CustMonVal	float64	Customer Monetary Value
ClaimsRate	float64	Amount paid by the insurance company in claims
PremMotor	float64	Annual Motor Premiums
PremHousehold	float64	Annual Household Premiums
PremHealth	float64	Annual Health Premiums
PremLife	float64	Annual Life Premiums
PremWork	float64	Annual Work Compensation Premiums

Table 1: Data and variable table

	count	mean	std	min	25%	50%	75%	max
FirstPolYear	10261	1991.06627	511.392431	1974	1980	1986	1992	53784
BirthYear	10259	1968.01891	19.711683	1028	1953	1968	1983	2001
EducDeg	10276	2.479077	0.795639	1	2	3	3	4
MonthSal	10241	2505.76731	1157.72948	333	1705	2500	3290	55215
GeoLivArea	10275	2.709781	1.266071	1	1	3	4	4
Children	10255	0.706972	0.455173	0	0	1	1	1
CustMonVal	10276	177.823446	1947.68338	-165680.4	-9.465	186.87	399.495	11875.89
ClaimsRate	10276	0.742958	2.919768	0	0.39	0.72	0.98	256.2
PremMotor	10242	300.492349	212.027094	-4.11	190.59	298.61	408.3	11604.42
PremHousehold	10276	210.55979	352.884514	-75	49.45	132.8	290.6	25048.8
PremHealth	10234	171.553063	296.658171	-2.11	111.8	162.81	219.04	28272
PremLife	10172	41.844886	47.488478	-7	9.89	25.56	57.79	398.3
PremWork	10191	41.277723	51.52853	-12	10.67	25.67	56.79	1988.7

Table 2: Descriptive table of the Dataset variables