Data Mining Project

MASTER DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS

CUSTOMER SEGMENTATION FOR A2Z INSURANCE COMPANY

Group W

Filipe Marçal Dias, number: r20181050

Inês Santos, number: r20191184

Manuel Marreiros, number: r20191223

January, 2023

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

INDEX

1. Introduction

The present report was developed under the scope of the final project of the Data Mining curricular unit of the Master's Degree in Data Science and Advanced Analytics at Nova IMS.

For the development of the project, we were presented with the customer data of a fictional insurance company, A2Z, with the goal of segmenting the database and finding relevant clusters of customers. The task at hand is quite important because a good customer segmentation will help the company to better serve and retain the existing customers, but also to gain new ones with the use of personalized marketing plans based on the insights gained from the analysis.

2. Data Exploration

2.1. Initial Analysis

After importing the provided dataset into a variable called ***customers***, we tried to better understand the data we were dealing with. This original dataset consists of 13 variables and 10 296 observations. In this section, we also identified the data types of the variables, as well as the existence of missing values and duplicate records. The duplicate records, however, most likely correspond to different customers who happen to share the same information, and their customer ID, which we defined as the table index, is different, so we decided to keep them.

2.2. Visual Exploration

For the visual exploration section, we first classified our features as either metric or non-metric. Immediately thereafter, we produced a set of histograms and a set of boxplots for each metric feature, which allowed us to visually identify significant outliers in all of them. Finally, we produced a Pearson correlation matrix, through which we concluded that the variables Salary and Birthday Year, and Customer Monetary Value and Claims Rate are strongly negatively correlated, particularly the last two, which show a correlation of **-0.99**.

3. Data Preprocessing

3.1. Encoding

For the variable related to the education degree of the customer, we made the decision to identify each unique value by a number from 1 to 4, in ascending order (Basic, High School, BSc/MSc, Phd).

3.2. Coherence Checking

Prior to any further analysis, we needed to make sure that our data made sense. For instance, we discovered that at least 2160 customers signed their first policy prior to being born. In fact, both the variables *BirthYear* and *FirstPolYear* by themselves presented abnormal values, such as a birth year corresponding to 1028, and a first policy year corresponding to 53 784, which are clear mistakes.

Additionally, we wanted to find customers who may have been overqualified for their age, for example, someone younger than 18 with a bachelor's or master's degree, of which we found none.

Finally, since the minimum legal age to work in Portugal is 16, we checked for records where people younger than 16 received a monthly salary, and where people between the ages of 16 and 18 received an abnormally large salary (for instance, superior to 1000€ per month). While we found records for both cases, all of them corresponded to customers who were born after the year of their first policy, so we inferred that was the main issue.

Wanting to preserve as much data as possible, our initial attempt to deal with this problem was by transforming the *BirthYear* values in the incoherent records into *None*, to then impute them using the KNNImputer. We reasoned that, since the variable *BirthYear* is dependent on the information that the customer provides, contrary to *FirstPolYear*, it would be more prone to errors. However, after double checking, we observed that the error persisted. Thus, we devised an alternative solution based on creating a new variable corresponding to the difference between the year of the customers' first policy and their birth year, *Diff_Birth_Policy*. As expected, this variable presented null values in the exact same records as *BirthYear*. We then imputed the null values of this new variable using the KNNImputer once again. For the missing values in *BirthYear*, we calculated them by subtracting this new variable to *FirstPolicyYear*. This way, we eliminated the issue, while also guaranteeing that the computed data made sense relatively to the neighboring observations.

3.3. Outlier Removal

After we were done with the coherence checking, some features still presented significant outliers.

In that sense, we first tried to remove the outliers using only the IQR Method, however, this resulted in 16.77% of the observations being deleted. Not wanting to delete such a large portion of our data, we opted for removing the outliers manually, with the aid of the boxplots produced in the visual exploration section. This way, we ended up with 99.64% of the original observations being preserved.

Immediately thereafter, we checked the boxplots once more, and saw that they had far less data points after the lower and upper whiskers, meaning we were able to reduce the outliers in a significant way.

3.4. Filling Missing Values

Analyzing the missing values, we concluded that almost all variables had some. In that sense, we opted to address them in different ways, according to the way in which they most likely occurred.

For starters, we though that the missing *FirstPolYear* values were most likely errors, since this variable should be created automatically whenever a customer does his first insurance policy. For that matter, we dropped all records where this value was missing, which accounted for a total of 30 rows.

Moreover, for non-metric features, we utilized the modes to fill the missing values. This was the case for *EducDeg*, *GeoLivArea* and *Children*.

Then, for the Premiums (*PremMotor*, *PremHealth*, *PremLife* and *Premwork*), we opted to convert the missing values to zero, since they most likely meant that the customer didn't spend any money on those services.

As for the remaining metric features, we opted for utilizing the KNNImputer, a widely used method that uses the k-Nearest Neighbors (5, in our case) to calculate a probable value for the data that is missing based on the most similar records. As we explained before, one of the variables in which we used this method was the *Diff_Birth_Policy,* that served as an auxiliar variable to get coherent values for the *BirthYear.* Once we were done doing the imputation, we checked once more if we had customers who had been clients for longer than they had been alive, and that was no longer the case.

3.5. Feature Engineering

In order to extract more valuable information from our data, we created the following features to add to ***customers***:

- **Age:** The age of the customers, obtained from subtracting *BirthYear* to *current_year (2016)*.
- **Years_as_customer:** The years passed since the customer's first policy.
- **Total_Annual_Premiums:** The sum of the amount of all Premiums in 2016.
- **Annual_salary:** The annual salary of each customer. Obtained by multiplying the monthly salary by 12.
- **Amount_paid_insurance_company:** The value paid, in two years, by the insurance company to each customer. In this feature we assumed that we had two equal years in terms of total annual premiums. This assumption was necessary since *Claims_Rate* is a 2 year variable. Then, we calculate the result by doubling the *Total_Annual_Premiums* and multiplying it by the *Claims_Rate.*
- **Amount_paid_year_insurance_company:** The value paid, in one year, by the insurance company to each customer. Since the previous feature was calculated assuming 2 years, we then divided *Amount_paid_insurance_company* by 2 to have an annual feature.
- **SalarySpent_Ratio:** The ratio of salary spent by each customer on insurance products. This variable is obtained by dividing the *Total_Annual_Premiums* by the *Annual_Salary.*
- **Annual_profit_customer:** This feature represents the annual profit that the customer generates to the company. It's obtained by subtracting the *Amount_paid_year_insurance_company* to the *Total_Annual_Premiums*.

Besides that, since we noted through some histograms that the distribution of three variables was heavily right-skewed, we decided to apply a **power transformation** to them, so that they would look more similar to a Gaussian distribution. In that process, we chose the method 'yeo-johnson', because it works well with both positive and negative values (which we had, since premiums could be negative). All in all, the values we created from the skewed ones were:

- **PremHousehold_PTransform**
- **PremLife_PTransform**
- **PremWork_PTransform**

3.6. Data Normalization

For the correct functioning of Data Mining algorithms, it is essential that our features are measured in a common scale, meaning that their range should be normalized. Additionally, the scaling method that is used should be chosen carefully, since it can have a serious impact on the output of our model.

Taking the aforementioned points into account, we tried two different scaling methods: **MinMaxScaler** between 0 and 1 and **StandardScaler**. We opted for settling with the MinMaxScaler method, since it produced higher r2 scores and more visually coherent clusters when we tested our models. The only exception for this were the self-organizing maps, which were acting in a weird way with the data normalized between 0 and 1, so we decided to switch to the StandardScaler for those.

3.7. Redoing Data Exploration

After all the pre-processing steps were done, and to make sure our data was well-prepared for the modelling part, we decided to call the ProfileReport function, that gave us a summary of our data after all the steps applied to it during the pre-processing. That way, we were able to assure that everything was fine, and all the mistakes were taken care of. We also made a pairwise relationship plot and another Pearson correlation matrix (this time already with the newly created variables) to reinforce the search for possible incoherences.

4. Modelling

4.1. Segmentation

With the exploration and preprocessing of the data out of the way, it was time to decide which perspectives we wanted to implement so that we could start applying the clustering algorithms. This step is of the utmost importance, since the quality of any cluster analysis depends on the variables used. We opted for utilizing two different perspectives based on:

- **Customer Value:** Evaluates groups of customers in terms of revenue generated and the costs of maintaining relationships with them.
- **Product Usage:** Divides the customers based on their needs, attitudes, and interests, taking into consideration their product usage.

For the first perspective, the variables used were: *Annual_Salary*, *CustMonVal*, *Years_as_customer*, *SalarySpent_Ratio* and *ClaimsRate*. As for the second one, we went with *PremMotor*, *PremHousehold*, *PremHealth*, *PremLife*, *PremWork*, *Total_Annual_Premiums*, *PremHousehold_PTransform*, *PremLife_PTransform*, *PremWork_PTransform*.

After selecting the variables that would be used in each perspective, we tried several combinations/subgroups of those variables to see which one would produce the best results. When choosing the combinations of variables to try, we made sure not to include correlated variables together, since that meant we would be inserting redundant information in our models and would have caused the quality of our clusters do decrease. For instance, *CustMonVal* and *ClaimsRate* were never used together, since they had a correlation of **–0.92**, which means that if we were to use both, the effect of one would cancel out the other. For obvious reasons, we also never included both a Prem variable and its respective transformed variable in the same subgroup. On contrary, since the *Years_as_customer* variable wasn't correlated with any other variable, we ended up using it in all the customer value tested subgroups, with this decision being validated when we later assessed feature importance.

That being said, we ended up with three subgroups for each perspective, that we then tested using the following models. For each subgroup we created a data frame with the method used, number

of clusters, r2 score and silhouette score, which then allowed us to evaluate which subgroup would be best for each perspective. The results for the best subgroup can be seen in the annex, both for the customer value perspective, and the product usage perspective.

4.2. Hierarchical Clustering

In (agglomerative) hierarchical clustering methods, each data point is treated as a separate cluster (singleton), and, by computing the proximity between each cluster, the algorithm will merge each pair of closest clusters, and then compute the new proximity amongst the remaining clusters. This process is repeated, eventually establishing a hierarchy of clusters.

When doing hierarchical clustering, we can opt from a variety of linkage methods, that is, the way we calculate the distances between clusters. In that sense, we got the r2 scores for 1, 2, 3, 4 and 5 clusters with four different linkages methods, **ward**, **complete**, **average** and **single**, concluding that the one that presented the best scores for the different number of clusters was ward.

Once we got the ideal linkage method for our purpose, we had to choose on the number of clusters we were going to create. To do that, we plotted a dendrogram (already with a Ward's method implementation) and then went ahead and used the number of clusters suggested by it.

4.3. K-means

When it comes to partition methods, we can choose between **K-Means** and **K-Medoids**. The difference between the two methods is that in K-Means each cluster is represented by a centroid that is the average of the points that belong to it, whereas in K-Medoids each cluster is represented by one of the points located near the center of cluster. In that sense, when computing the r2 scores for each of the methods, we understood that it would be better to go with K-Means for our implementation. In fact, we actually ended up using a small variation of K-Means, called K-Means++, which is pretty much equal to K-Means, with the only difference being the fact that it has a smarter seed initialization, which is one of the biggest flaws on K-Means: the fact that it is dependent of the initial random position of the seeds.

Then, and although we already had an idea of the number of clusters we would make from the hierarchical clustering done before, we calculated the inertia for different cluster sizes. Once we got the inertias for the different possible sizes, we plotted it to an inertia plot, also known as Elbow-Method. Then, looking at the point of inflection in the plot, we were able to infer what would be an adequate number of clusters to use.

To complement the analysis made with the inertia, we also calculated and plotted the different silhouette scores with the help of a function that received a data frame as input, and returned the average silhouette scores (and respective plots) for different numbers of clusters. We then looked for the value of k that gave us the highest silhouette score, since a high value indicates that the objects are well-matched to their own clusters and poorly matched to neighboring clusters, and used it to create the cluster.

4.4. Mean-Shift

Mean-Shift clustering is a sliding-window-based algorithm that attempts to find dense areas of data points (areas where there is a high concentration of points). It works by shifting said window to the position of the mean of the points that are within the window radius, until we reach a stopping condition (when moving to a new location won't accommodate more points than the ones in the current window).

In our implementation, we started by estimating the bandwidth, which is the most important parameter in Mean-Shift Clustering. To do that, we used a function from the sklearn cluster module that uses the mean distance between points to estimate the bandwidth. We then played with the quantile parameter in order to get an adequate number of clusters, trying different possibilities and seeing what the outcome would be. With the right parameters, mean-shift ended up producing interesting results.

4.5. DBSCAN

DBSCAN is a density-based clustering algorithm that model's clusters as dense regions in the data space, separated by sparse regions with few to no points.

The two most important parameters in DBSCAN are epsilon (ε) and min_samples (or minPoints). The former consists of the size of the radius that defines the neighborhood and the latter consists in the minimum number of points that have to be inside the neighborhood of a point for it to be considered a core point. After fine-tunning those parameters, with the help of a K-distance graph to find out the right eps value, we also used DBSCAN to detect some more outliers (noise) that may have been undetected in the earlier stages of our outlier analysis.

DBSCAN ended up producing the least interesting results of all models, which could be due to the fact that "density-based methods suffer from severe drawbacks when applied to large spatial databases" [(1)], which might be the case of our dataset.

4.6. Gaussian Mixture Model

The **Gaussian Mixture Model** is a distribution-based clustering algorithm, meaning that the model assumes the existence of various gaussian distributions, and that each represent a different cluster. By plotting the Akaike information criterion (AIC) and the Bayesian information criterion (BIC) - which are measures of model performance that account for model complexity – we were able to determine the number of components, which is a parameter equivalent to the number of clusters, and then use that to get our clusters.

4.7. Self-Organizing Maps

SOM are unsupervised neural networks that can be used for clustering and to visualize high-dimensional data in a lower-dimensional space, such as a 2D map.

In SOM, the data points are represented by nodes/units on the map, and the connections between the nodes are adjusted during the learning process. The map is typically initialized with a random set of connections, which we controlled with the initialization parameter. The connections are then updated through training. In our case, we initialized a 20x20 grid, meaning that the final map will have 400 units. During training, each data point was presented to the map and the nodes adapted to

represent them. The node that wins the competition is called the Best Matching Unit, and its connections are adjusted to be more similar to the input data. The connections of the nodes that are close to the winning node are also adjusted, but to a lesser degree. This process is repeated for each data point, and over time the map becomes more organized, with similar data points clustering together on the map.

Once we were done with the training process, we wanted to visualize the **component planes, U-Matrix and Hit-Map** to gain some more information on how our SOM was adapting to the data that was presented to it. For instance, with the Hit-Map we were able to see which units were deemed BMU the most times, and with the U-Matrix we were capable of having an idea of the clusters that could be formed.

4.7.1. Hierarchical Clustering Over SOM

Once we had our Self-Organizing Map, and the respective units, we tried to apply hierarchical clustering over them. To do that, we repeated the steps we had done in hierarchical clustering, but instead of inputting the data frame, we passed the nodes of the SOM matrix instead. We were then able to visualize the clusters we had created. It is important to note that this approach combines the strengths of both hierarchical clustering and SOMs, but it also has some limitations. One limitation is that the clusters produced by the hierarchical clustering algorithm may not align well with the topological structure of the SOM map. Additionally, the resulting clusters may be difficult to interpret, as they are based on the SOM units rather than the original data points.

4.7.2. KMeans Clustering Over SOM

The principle of KMeans over SOM is the same has the one applied when doing Hierarchical Clustering over SOM. We just created an object for KMeans, with a predefined number of clusters, and then fit and perform predictions over the SOM units.

5. Merging the Perspectives

After producing the individual clusters for the two perspectives (customer value and product usage), we decided to merge both perspectives to get a broader understanding of our data. To do so, we used the best subgroup of each perspective (customer_perspective2 and product_perspective3) and the labels of the best clustering algorithm for that segment (K-Means for both), which yielded in a data frame with the different cluster groups.

Since some groups had very few elements, it made no sense to keep them as individual clusters, so we used hierarchical clustering to merge the closest clusters, ending up with a more reasonable number of clusters (4) and all with a significant number of customers in them.

6. Cluster Analysis

Once we got the final clusters with the merged perspectives, it was time to analyze them, as well as the ones produced by the individual perspectives. Here, we looked at cluster sizes and relative values for each variable, with the help of histograms and line charts. The customer value perspective ended up with four equally divided clusters being the main differences in the variables *Annual_Salary*, where we clearly have two clusters with higher values (0 and 3), and *Years_as_customer*, also with two

clusters (1 and 3) with higher values that the others. The other variable, *ClaimsRate*, doesn't show any difference between clusters. Regarding the product perspective, we finished with five clusters, three with a higher absolute frequency and two with a smaller one. Moreover, we saw that clusters 0, 2 and 4 form a stable uniform-like distribution across all the variables, whereas clusters 1 and 3 appear to go to more extreme values, represented by higher and lower peaks.

At this stage, we also reintegrated the variables that were not used for clustering, namely the Education Degree, Living Area and Children. These categorical variables were not useful for the creation of the clusters, but they can still provide information about them, which is what we intend for them to do. In that sense, we calculated the mean of the three variables for each final cluster, and we also plotted violin charts to visually analyze them.

7. Cluster Visualization

While the cluster analysis can be helpful to understand the data, visually representing the clusters that have been identified in our data is always a good practice and can be beneficial for several reasons, namely the fact that it allows for a better communication of the obtained results, or the fact that it allows to evaluate the quality of the clusters and determine whether they are meaningful and interpretable or not. We can even go as far as to say that it can be useful for debugging and improving the clustering process, as it can help us identify problems, subtleties or issues with the clustering algorithm or the data itself that we were not able to detect in the cluster analysis stage.

In that sense, to visualize our multidimensional clusters in 2 dimensions, we started by using t-SNE (t-distributed stochastic neighbor embedding), a non-linear dimensionality reduction technique that works by minimizing the divergence between the probability distributions of the high-dimensional data and the low-dimensional data.

Then we went ahead used UMAP for the same effect. UMAP (uniform manifold approximation and projection) is also a nonlinear dimensionality reduction technique that is designed to preserve the structure of the data as much as possible. It works by constructing a low-dimensional representation of the data that preserves the distances between points in the high-dimensional space, and the greatest advantage that it has over t-SNE is that it tends to produce more stable results across multiple runs, whereas t-SNE results can vary quite a bit depending on the initialization and other parameters. (2)

8. Feature Importance

To finish off, we wanted to understand why our clusters were formed in the way they were, and which variables contributed the most to it. For that matter, we calculated the feature importance, which is a measure that refers to the degree to which a particular feature or set of features influences the assignment of data points to different clusters. In general, features with higher importance are likely to be more informative for clustering and may be more useful for understanding the underlying structure of the data.

To assess feature importance, we first calculated the r2 score of each variable. Through this analysis, we understood that the most important features were *PremMotor*, *Years_as_customer* and *PremHealth*. Then, we used a decision tree to see which features would be used first to classify our

instances, since those features are the most important ones. We ended up coming to the same conclusion as before, with the three referred variables being enough to classify around 90% of our customers correctly, which is pretty satisfactory.

9. Marketing Plan

We will now analyze the final four clusters that resulted from merging the two perspectives and propose marketing actions that can be followed to better satisfy each individual group of people.

Cluster 0: This cluster, which is the one with most observation in it (3521), can be looked at as the cluster of the "normal clients" since all their values do not differ much from the mean. The only exception is the variable *PremHealth*, that goes slightly above the mean, and the Annual Salary that is quite high. These people also have a high level of education, with most of them having completed either a bachelor or a master's degree.

Cluster 1: This cluster with 2206 people has the highest value for *PremMotor*, but the lowest one for the remaining premium types, meaning that we are probably looking at vehicle owners who want to protect themselves from incurring in any financial losses that may arise due to damage or theft of the vehicle. People in this cluster are very highly educated, being the one with the highest number of people with PhD's in it.

Cluster 2: This cluster has 3417 people, and it is very similar to cluster 0, with the only note-worthy difference being the fact that this cluster's customers have been using the company's services for a longer period.

Cluster 3: This was the less numerous cluster, with few more than 1000 people on it. It is characterized by people who do not have a high annual salary, but still spend a lot on the different types of products, with the exception being the variable *PremMotor*. In fact, people in this cluster are the ones who spend the most in four out of five types of insurance products and are the ones who spend the least money on vehicle related products. It is also noticeable that people who fall inside this cluster are less educated that the overall population, which might be the reason why they don't have such high salary.

Although each cluster has its particularities, all of them also have some things in common. For instance, in all clusters people who have children are more predominant than people who don't, which means that it can be a good idea to promote health insurances amongst these families that are looking for forms of protecting themselves. Not only that, most people live in geographical area number 4, which might indicate that the marketing should be mostly done at that location. That being said, our simplified marketing proposal consists of the following:

In **clusters 0 and 2**, considering that they have high annual salaries, the company should aim to make the people in these clusters spend more on the different types of products. This doesn't mean that the company should start bombarding them with offers, because it might make them unsatisfied or bothered, which is something that they most definitely don't want to do, since these 2 clusters comprise a very large part of the current customer base. In particular, cluster 2 has the people with the largest number of years as customer, so they might not like a sudden shift in the communication plan of the company. For these people, if it is not possible to make them spend more on insurance

products, the company can also suggest the investment on retirement savings plans, which would be very beneficial for the customers since they have high salaries and can probably save a good amount.

For **cluster 1**, we are looking at people who spend a lot on motor insurance, but are at the bottom in the other types of products. In that sense, we can reinforce their spendings on motor related insurances by making a package where they can get a discount for a motorcycle insurance if they already have one for the car. Furthermore, since these people most likely spend a lot of money fuelling their vehicles, it could be a good idea to do partnerships with gas stations, so that people can benefit from discounts on fuel if they have this insurance product. We can also apply cumulative discounts to incentivize them to diversify more the type of insurances they purchase. A proposal of a discount plan can be consulted in the appendix.

For **cluster 3**, we could explore the possibility of cross selling the motor insurance with other premiums, since the members of this cluster have shown little interest in this type of product. Nevertheless, the fact that this cluster shows the lowest average annual salary amongst the four clusters can be a barrier in trying to make these people buy more products, since they already spend a lot of money in other insurance types, specially Household, Life and Work. For that matter, applying discounts to the Motor products for people who already have other Premiums could be a good idea, specially for people in this cluster that don't earn a lot of money. Not only that, the fact that people in this cluster don't have high salaries and have less education than the general population, might indicate that they are quite young and still studying or just starting their professional careers. In that sense, if the company does decide to apply the suggested discount, it could be a good idea to advertise for it near universities or other areas of interest for younger generations.

Lastly, it could also be a good idea for the company to create a loyalty card or even a referral program, where someone would get rewarded for inviting a friend that ends up making an insurance.

10. Conclusion

Through the development of this project, we were able to both consolidate the materials learned during the Data Mining course, as well as apply knowledge obtained through self-study. Although challenging, it was enriching, allowing us to discuss and investigate certain topics more thoroughly. In fact, we can even say that we went as far as learning more about insurances to obtain some domain-specific knowledge, which was helpful for some parts of the project.

The project was not, however, without its setbacks. A major concern we had was regarding the scaling of the data, because we were not aware of how much the scaling method would influence the performance of the algorithms. It was not only until we tried different scaling methods that we understood that and concluded that MinMax worked better for most algorithms, with the exception being the Self-Organizing Maps.

Finally, we can say that we are quite satisfied with the obtained results, since we believe that we were able to uncover meaningful and relevant nuances present on the data. The clusters we obtained seem coherent and are supported by good scores and a thorough examination, so if the company was to use them, they could probably make great use of the analysis produced to satisfy exiting customers and gain new ones.

11. REFERENCES

(1) - Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (n.d.). 1996. *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. Institute for Computer Science, University of Munich.

(2) – Becht, E., Dutertre, C., Kwok, I., Ng, L., Ginhoux, F., Newell, E. 2018. *Evaluation of UMAP as an alternative to t-SNE for single-cell data*

12. APPENDIX

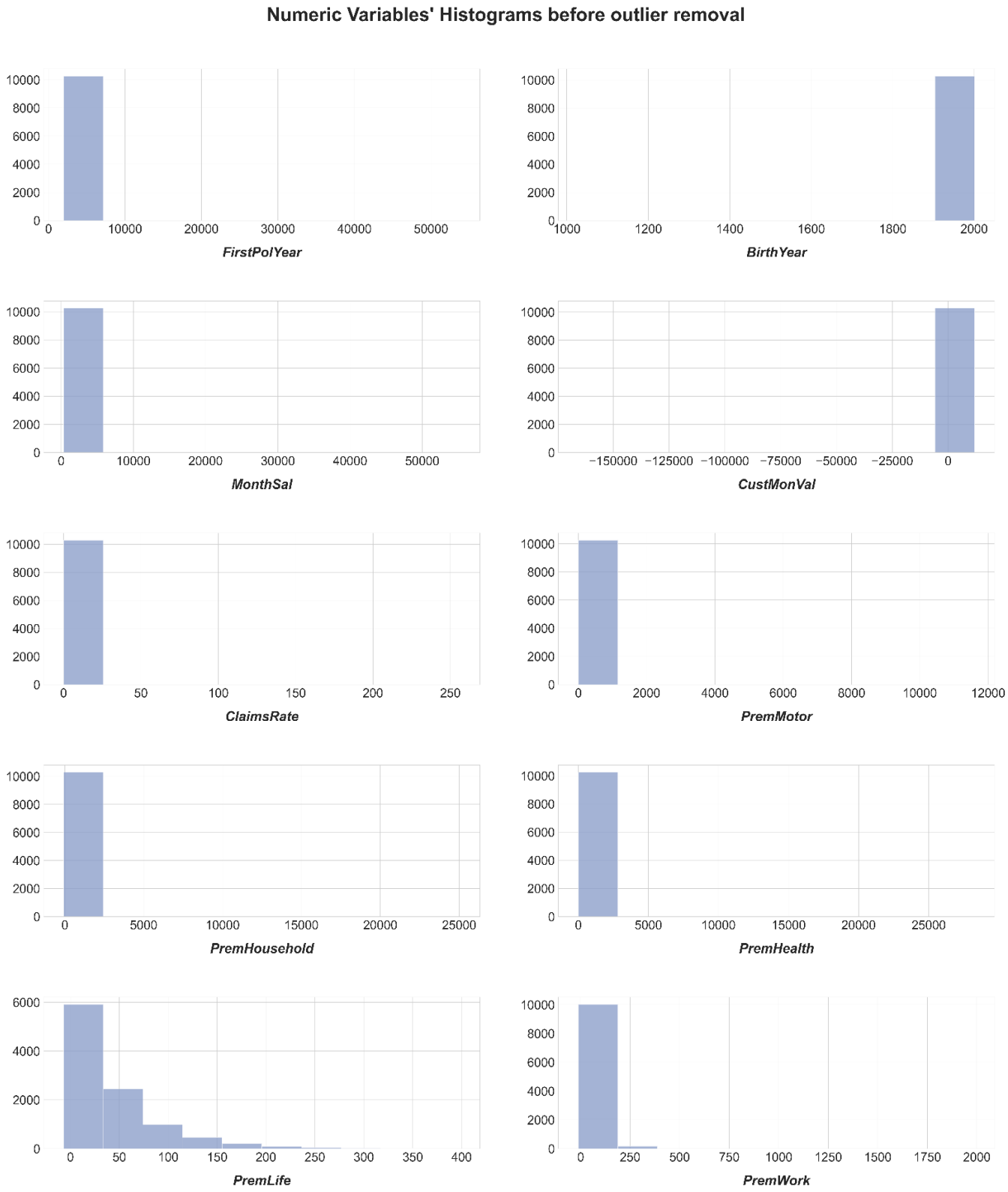

Figure 1 - Numeric Variables Histograms before outlier removal

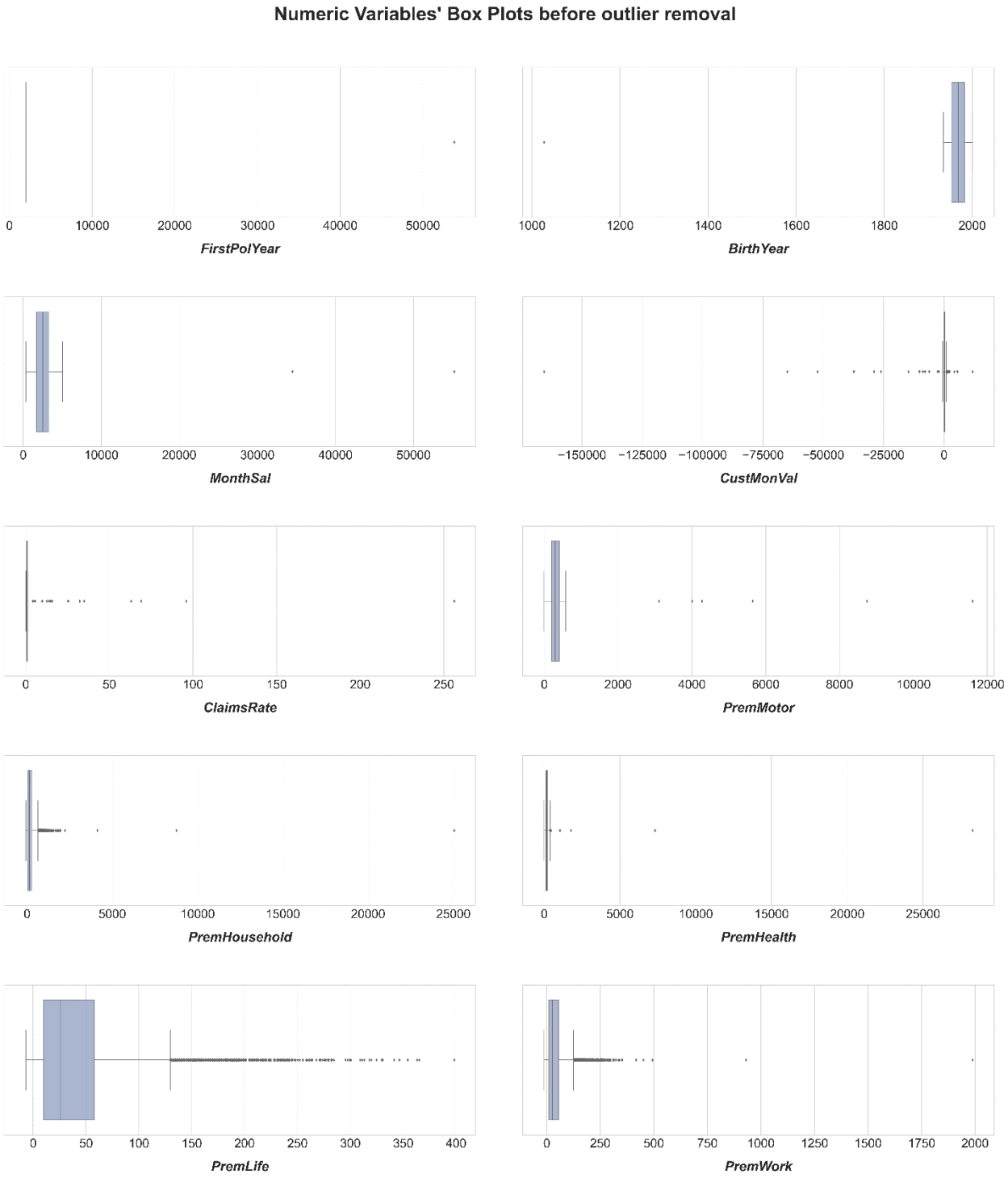

Figure 2 - Numeric Variable's Boxplots before outlier removal

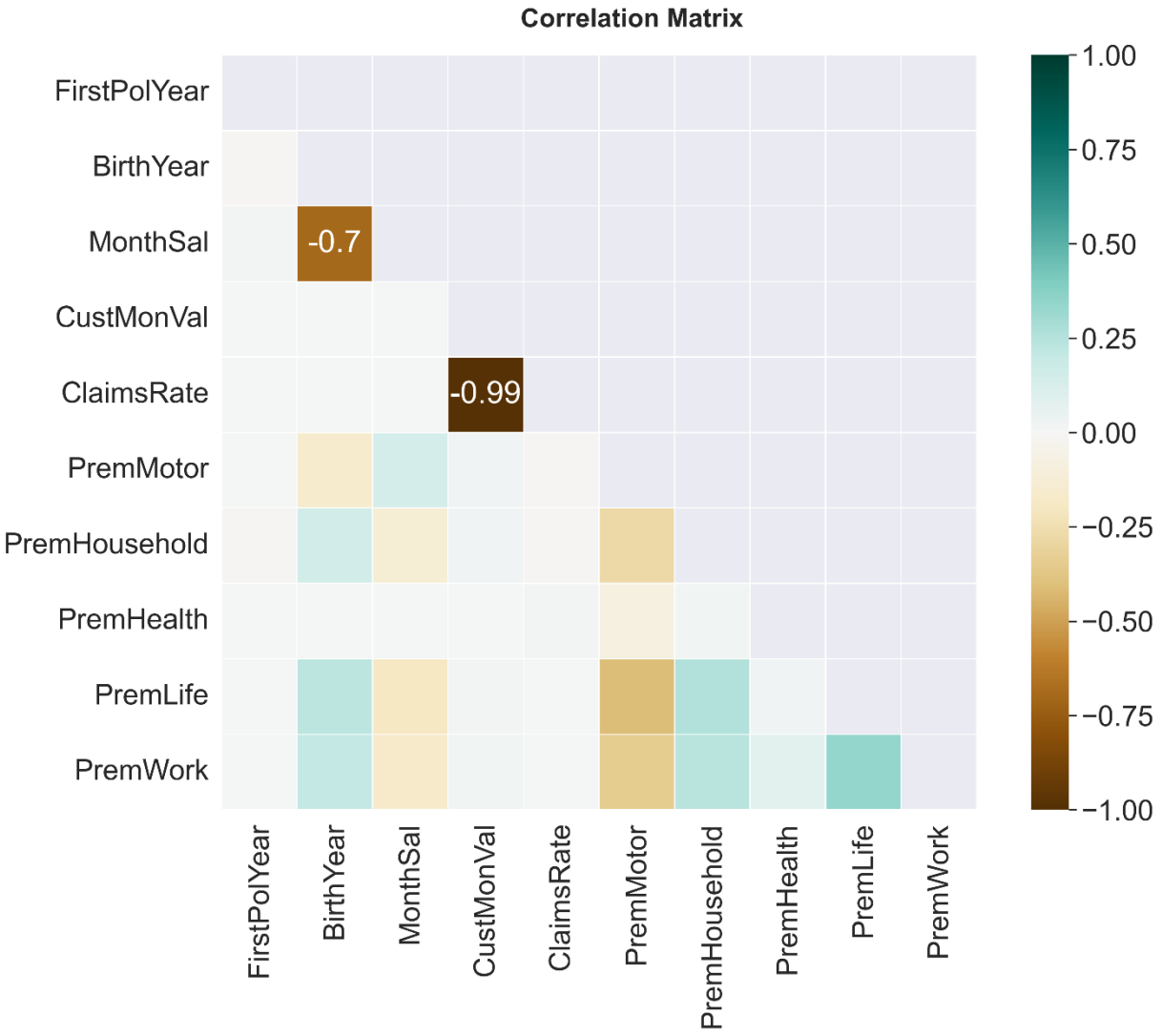

Figure 3 - Correlation Matrix

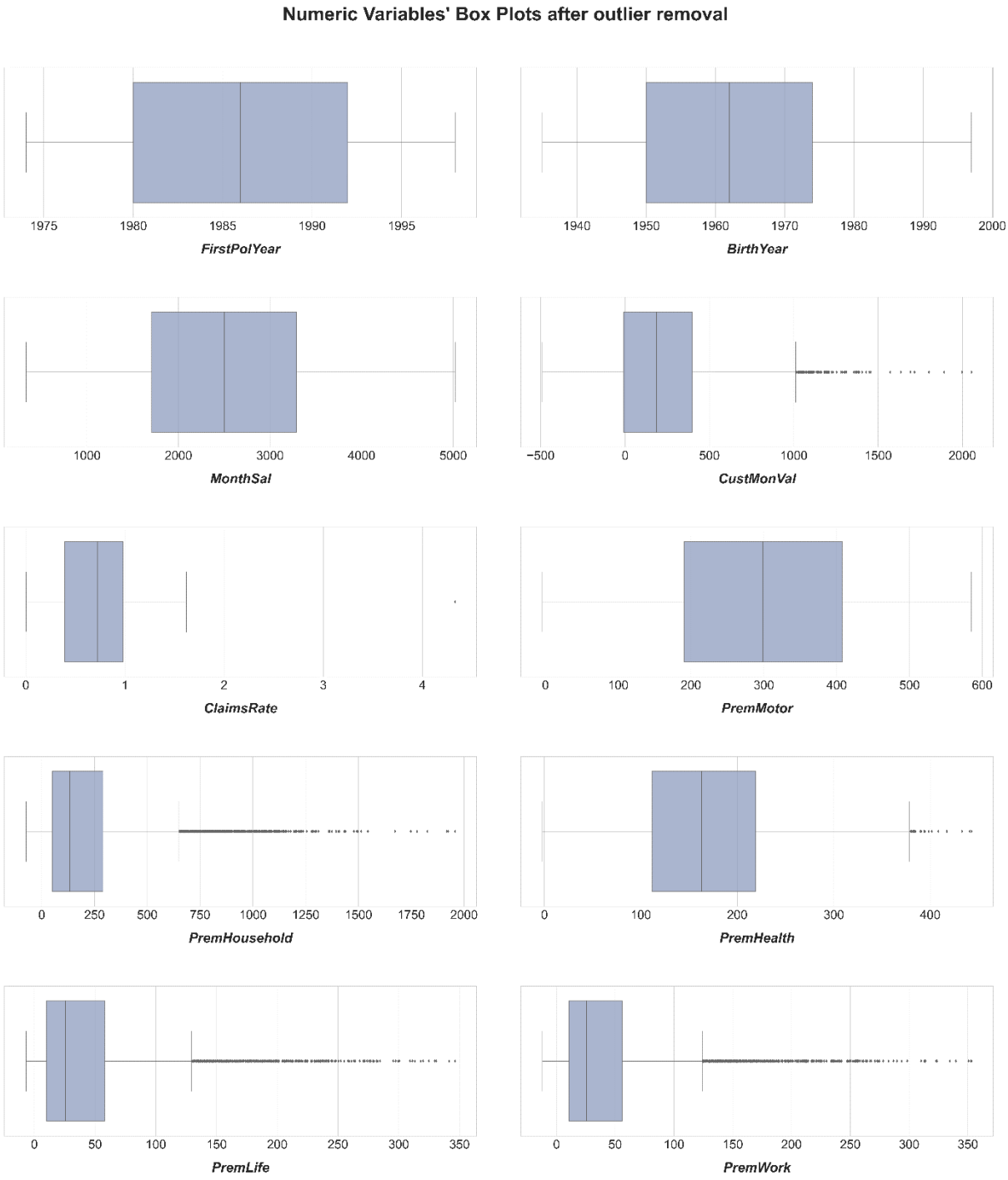

Figure 4 - Numeric Variables Boxplots after outlier removal

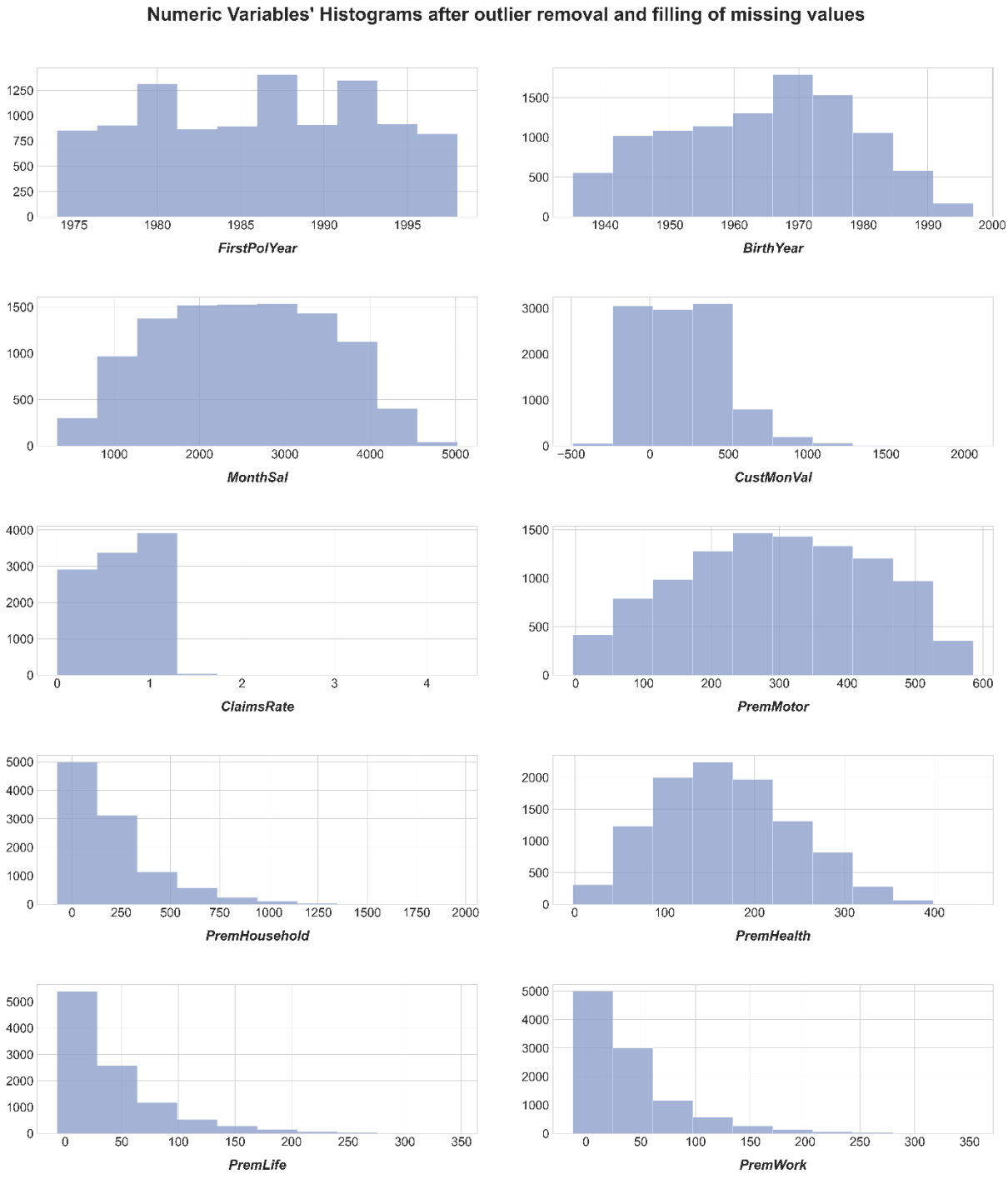

Figure 5 – Numeric Variables Histograms after outlier removal and filling of missing values (noticeable skewness in *PremHousehold*, *PremLife* and *PremWork*)

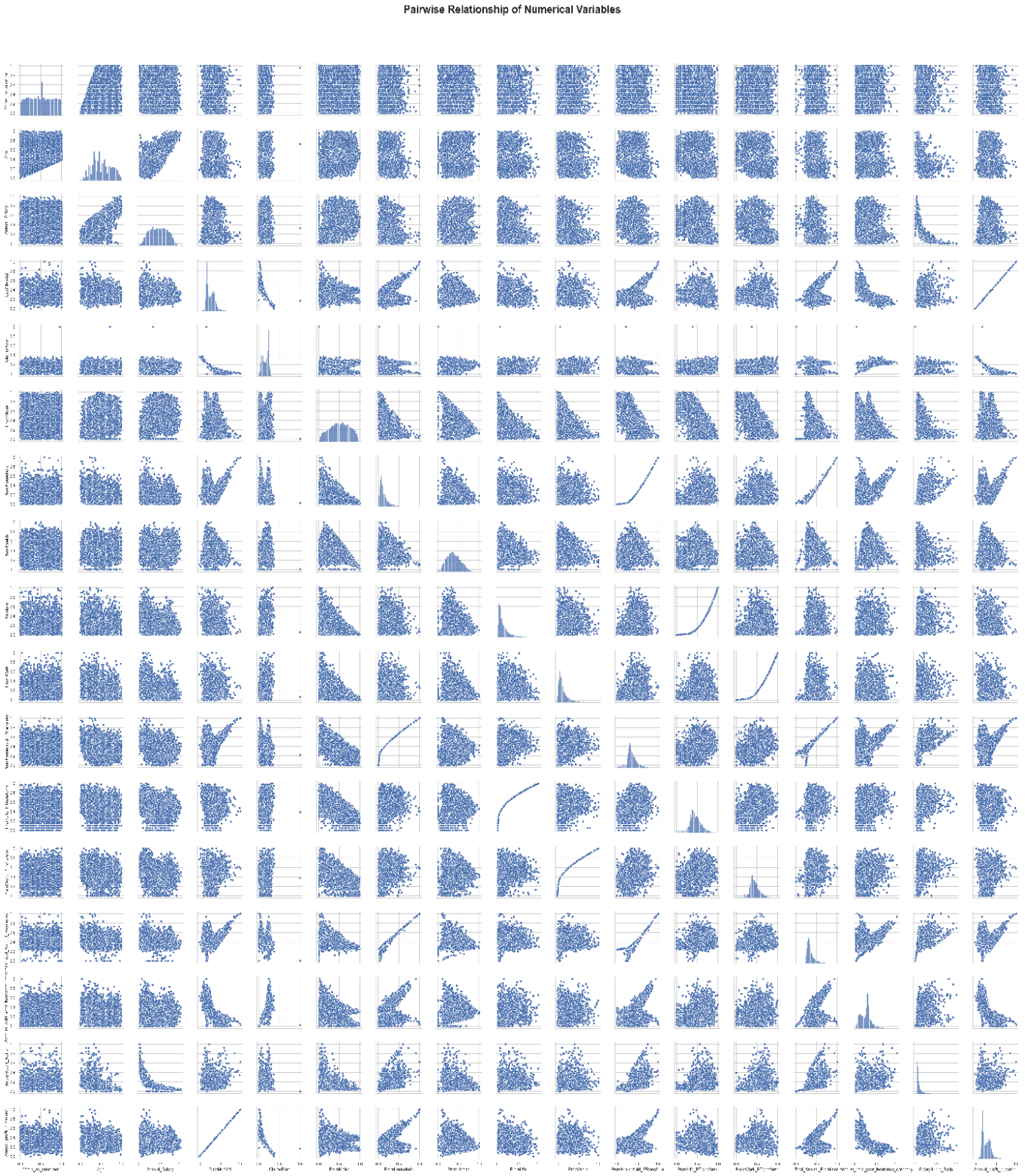

Figure 6 - Parwise Relationship between all numeric variables (including the newly created ones)

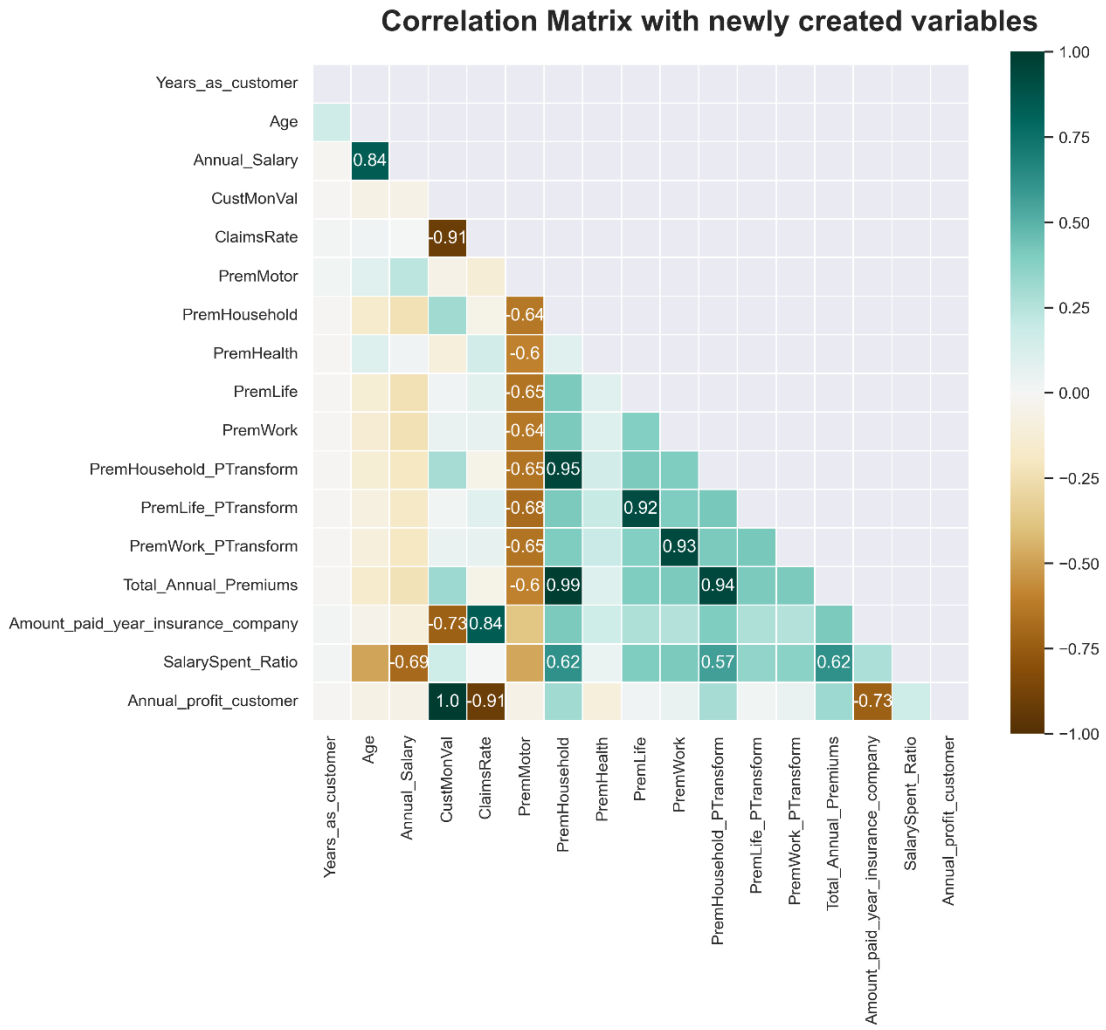

Figure 7 - Correlation Matrix with the final variables

Method	Clusters	R*2	Silhouette
Hierarchical	4	0.6341	0.2639
K-means	4	0.7041	0.3408
Mean-Shift	4	0.5682	0.3182
DBSCAN	2	0.0004	0.2768
GMM	4	0.7021	0.3396
Hierarchical over SOM	4	0.4714	0.2207
K-means over SOM	4	0.5049	0.247

Table 1 – R2 and Silhouette scores for best subgroup of the Customer Value Perspective

Method	Clusters	R*2	Silhouette
Hierarchical	5	0.6631	0.1961
K-means	5	0.6914	0.2508
Mean-Shift	4	0.374	0.3435
DBSCAN	3	0.0158	0.2746
GMM	5	0.4943	0.1104
Hierarchical over SOM	5	0.5795	0.1753
K-means over SOM	5	0.5944	0.1893

Table 2 - R2 and Silhouette scores for best subgroup of the Product Usage Perspective

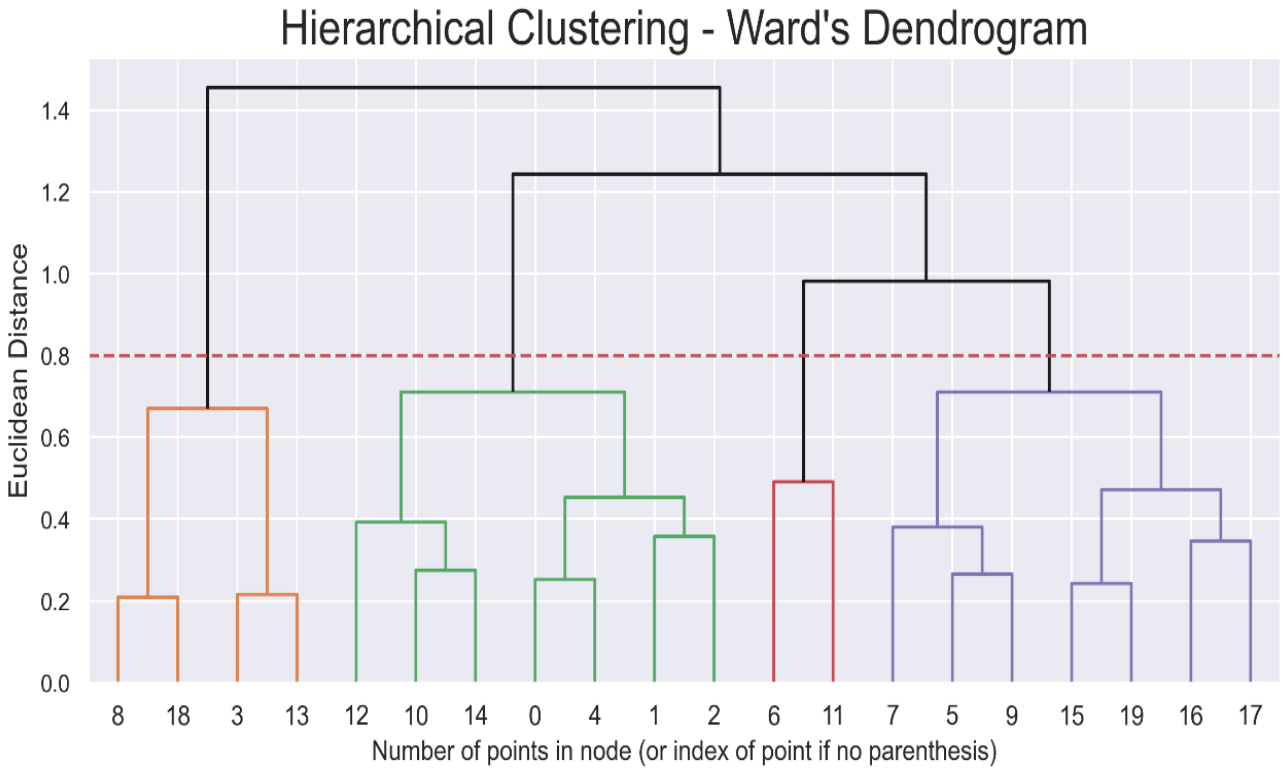

Figure 8 – Dendrogram for the merged perspectives

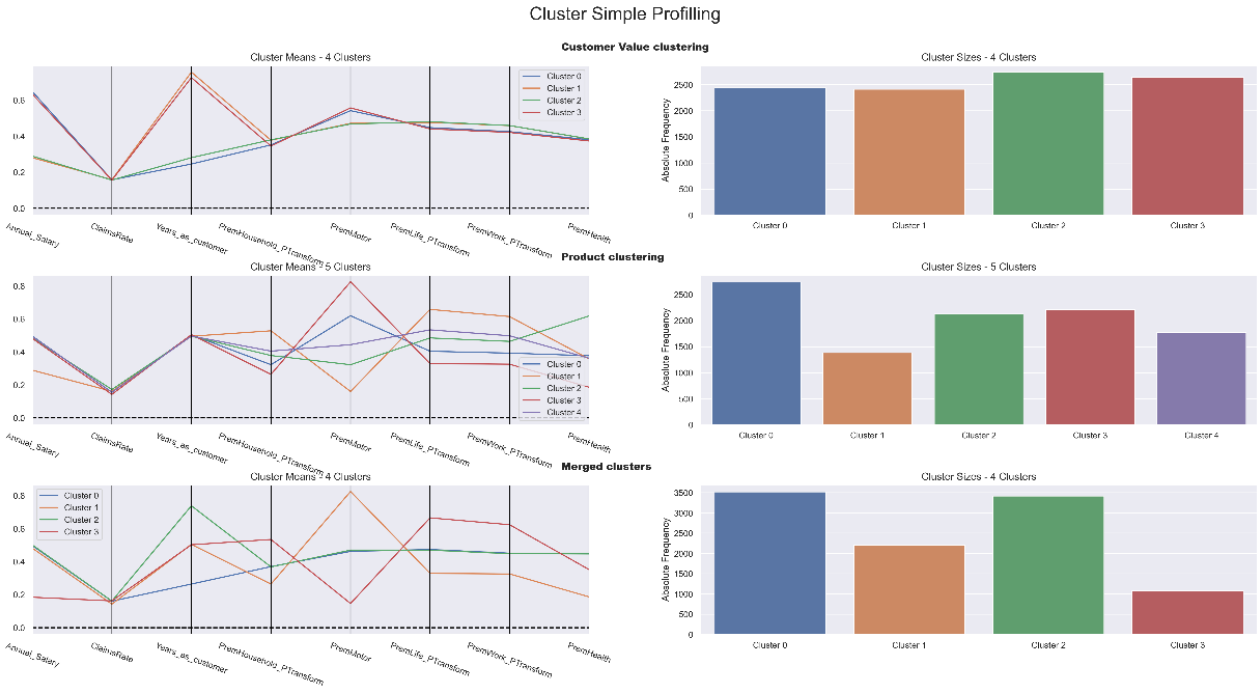

Figure 9 – Cluster Profiles for individual perspectives and merged clusters

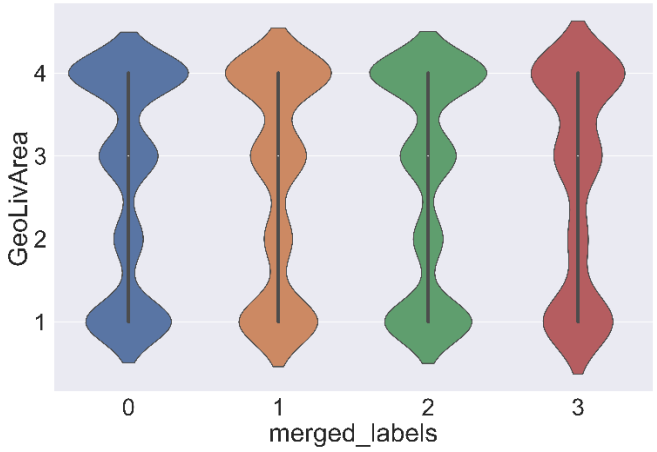

Figure 10 – Violin Plot for *GeoLivArea*

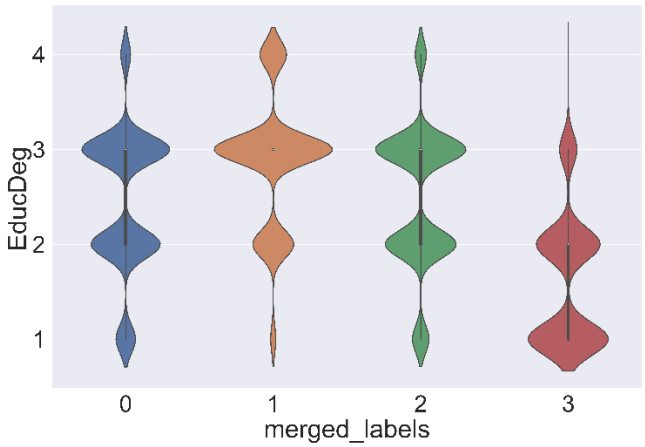

Figure 11 – Violin Plot for *EducDeg*

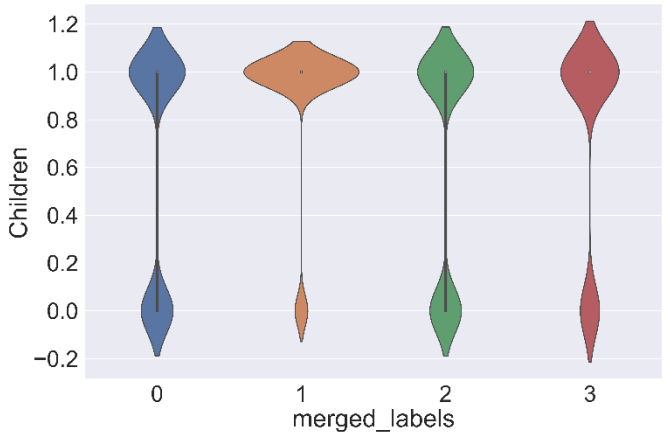

Figure 12 – Violin Plot for *Children*

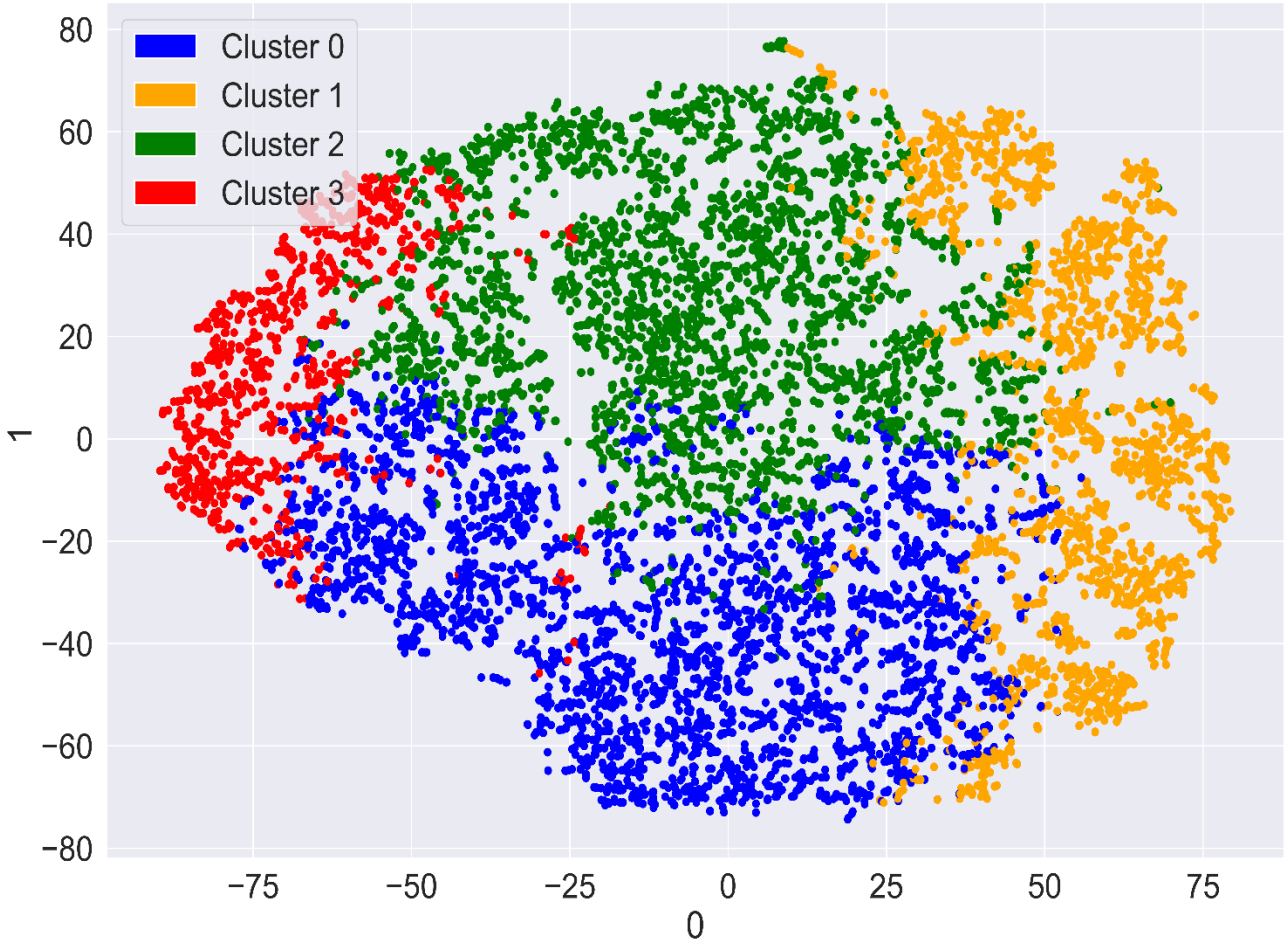

Figure 13 – t-SNE Visualization

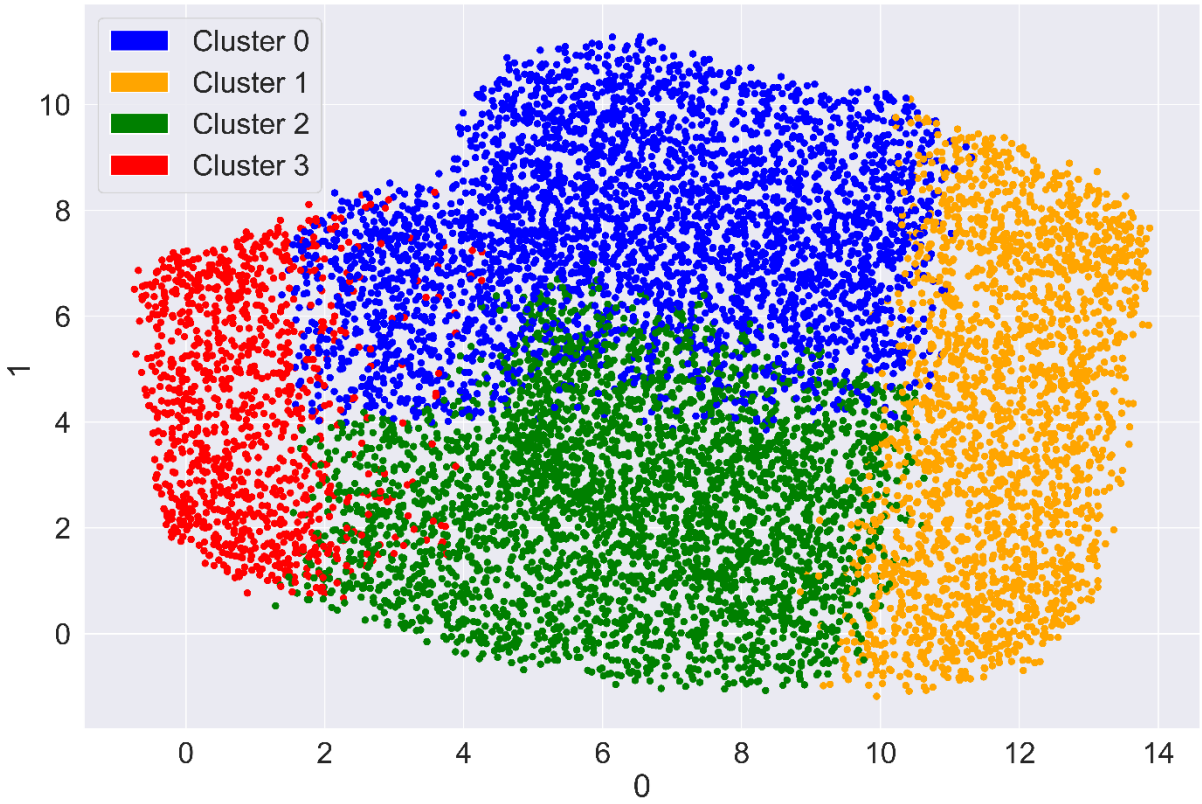

Figure 14 – UMAP Visualization

Features	Importance based on R*2	Importance based on the decision tree
Annual_Salary	0.2092	0
ClaimsRate	0.0104	0
Years_as_customer	0.502	0.4741
PremHousehold_PTransform	0.3633	0
PremMotor	0.6616	0.5204
PremLife_PTransform	0.3929	0
PremWork_PTransform	0.3615	0
PremHealth	0.3984	0.0055

Table 3 - Feature Importance

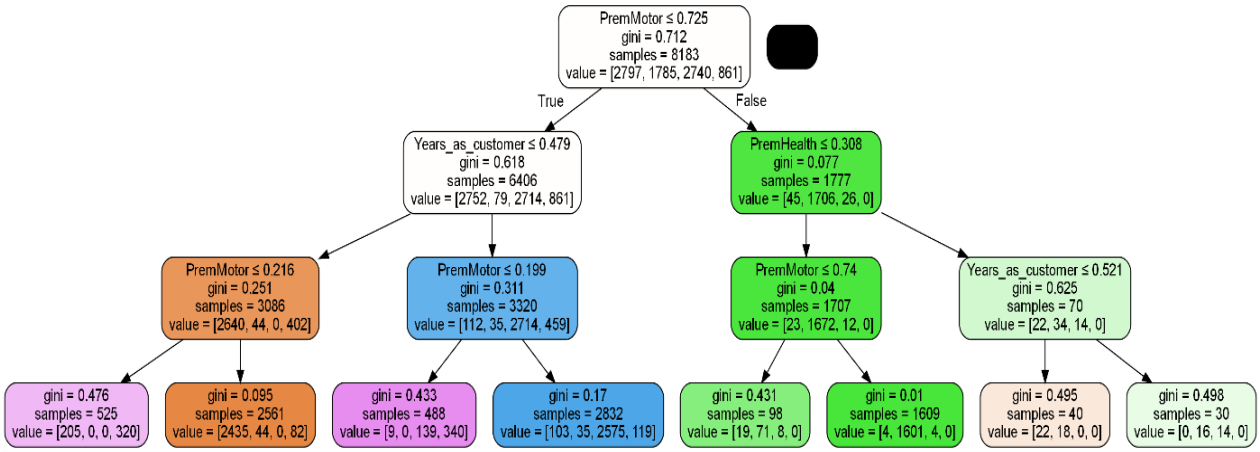

Figure 15 – Decision Tree used to calculate Feature Importane

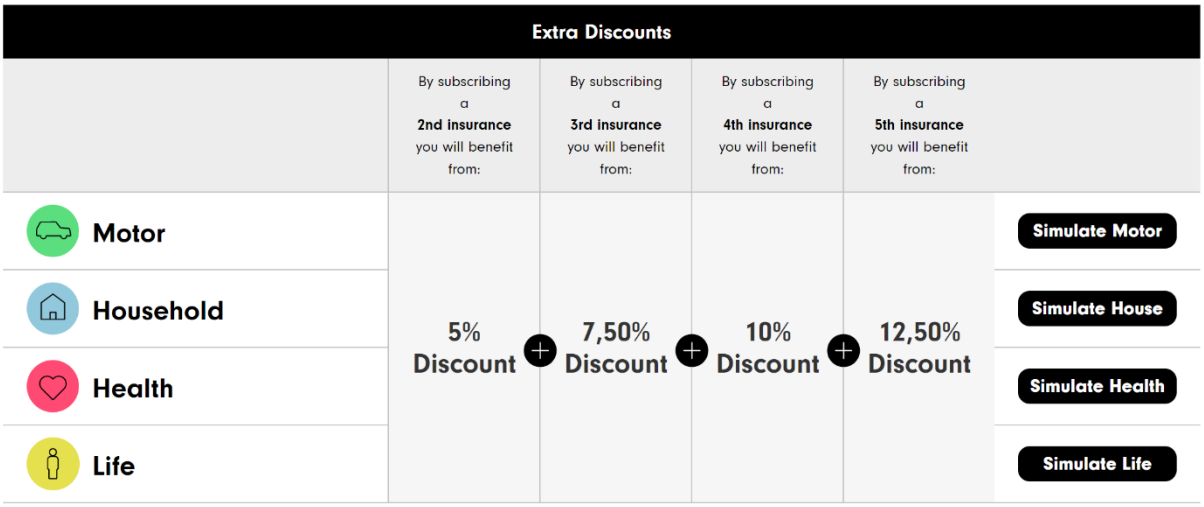

Figure 16 – Discount Plan (based on logo.pt)