

**Data Mining**

Final Project

Alex Anthony Panchot (M20190546)

Hugo Saisse Mentzingen da Silva (M20190215)

Rennan Valadares Ornelas Araújo (M20190146)

# Table of Contents

[Table of Contents 2](#_heading=h.30j0zll)

[1. Introduction 3](#_heading=h.1fob9te)

[2. Preprocessing 4](#_heading=h.2xcytpi)

[2.1. *NaNs* and outliers 4](#_heading=h.3znysh7)

[2.2. ‘Age’ column assessment 4](#_heading=h.2et92p0)

[2.3. Categorical features classification 7](#_heading=h.1t3h5sf)

[2.4. Numerical features regression 8](#_heading=h.17dp8vu)

[3. Analysis 9](#_heading=h.1ci93xb)

[3.1. Features correlation 9](#_heading=h.3rdcrjn)

[Quantile Matrix 10](#_heading=h.3whwml4)

[Boxplots 10](#_heading=h.2bn6wsx)

[Predictive Modeling 11](#_heading=h.1y810tw)

[Summary 12](#_heading=h.4i7ojhp)

# Introduction

This report addresses the final project in the Data Mining course of Master Degree Program in Data Science and Advanced Analytics of Nova IMS. The groups of students received a fictional insurance company database, containing personal customers’ data as well as insurance consumption information. The features contained in the dataset are described in the [Project Description](http://elearning.novaims.unl.pt/moodle/pluginfile.php?file=%2F53291%2Fmod_folder%2Fcontent%2F0%2FProject%20Description.pdf&forcedownload=1) document.

While the team impersonated a data mining consultancy, the final objective of this work was to “develop a Customer Segmentation in such a way that it will be possible for the Marketing Department to better understand all the different Customers’ Profiles”.

Therefore, we present in the following pages the process applied for this dataset treatment, the questions, findings, discussion, and recommendations that arose within it.

This report is accompanied by the file [Data\_Mining\_Project.ipynb](https://1drv.ms/u/s!AvRbdQNKauSThrNiHocr8C_nsKQyfg?e=fDvik3) which contains all the Python code used.

# Data Preparation

The dataset was initially converted into a Pandas DataFrame and the existing year-based columns (Brithday Year and First Policy´s Year) were rescaled to facilitate the interpretation, being substituted by Age and First Policy´s Age. For further analysis, the Premium: Sum feature was also added to the insurance\_df DataFrame, representing the sum of all premiums paid by the customer.

# *NaNs* and outliers

The initial assessment revealed 309 rows (3% of the total) with at least one blank cell (*NaN*) as well as the existence of some outliers or even noisy data (Figure 1). Instead of merely discarding the rows with NaNs, which could result in a significative loss of information, the team decided to apply techniques to estimate numerical and categorical features.

These techniques, regression and classification, respectively applied to numerical and categorical columns, first needed the removal of outliers, since it would provide better estimations.

**Outlier treatment:** since the dataset is not normally distributed, the team used the *interquartile range* (IQR) as the method to remove outliers. The usage of 1.5 IQR as the initial criteria for all columns revealed that some features had a large amount of observations to be considered as outliers (Household, Life and Work Compensations with more than 600 outliers each). In these especific cases, only the most extreme outliers were removed (above 3 IQR).

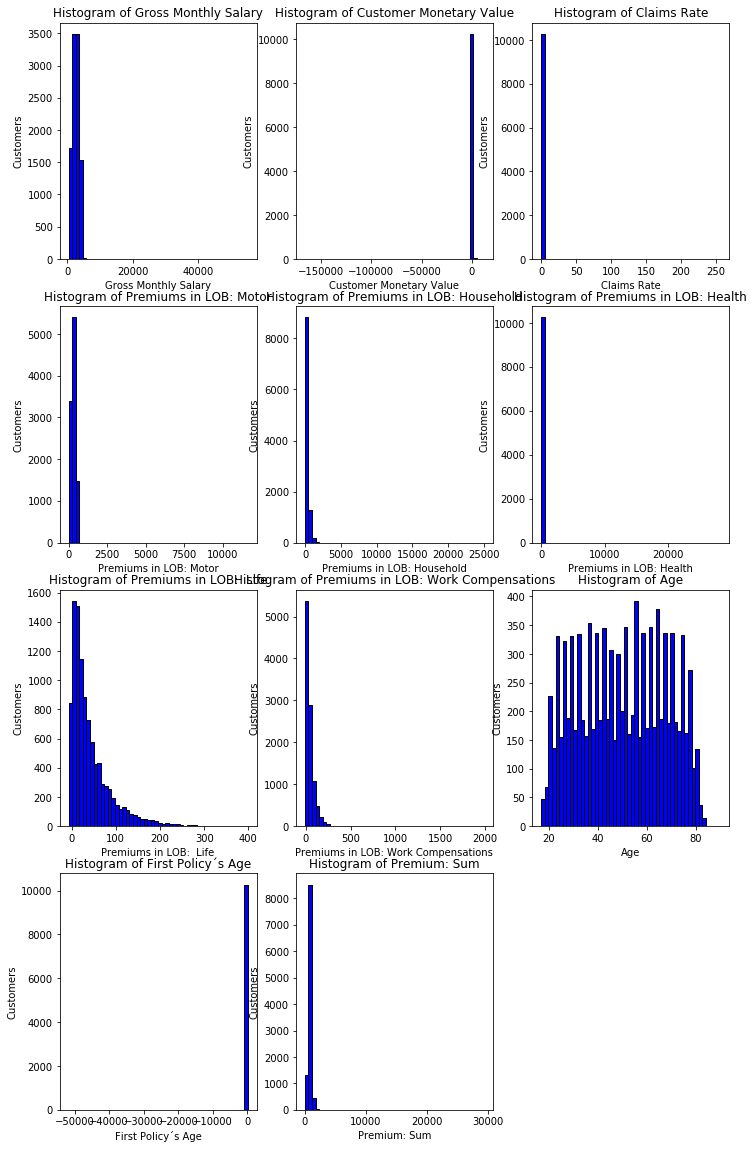
The criterion above resulted in at most 162 outliers for the same feature and 439 outliers in total (4.3% of the dataset) that were temporarily removed from insurance\_df. The resulting data is presented in Figure 2.

Rows with more than 3 NaNs were sough considering they could also undermine the estimators. However, there were no rows suiting this criterion.

# ‘Age’ column assessment

The visual analysis of the Age feature motivated a deeper assessment regarding its reliability, considering many customers have Age lower than First Policy´s Age. In fact, 1702 customers are in this situation and, additionally, 77.3% of customers below 19 years old already have children, which is quite uncommon.

For this reason, Age was not considered in subsequent analyses.



*Figure 1 - Features histograms before removing outliers*

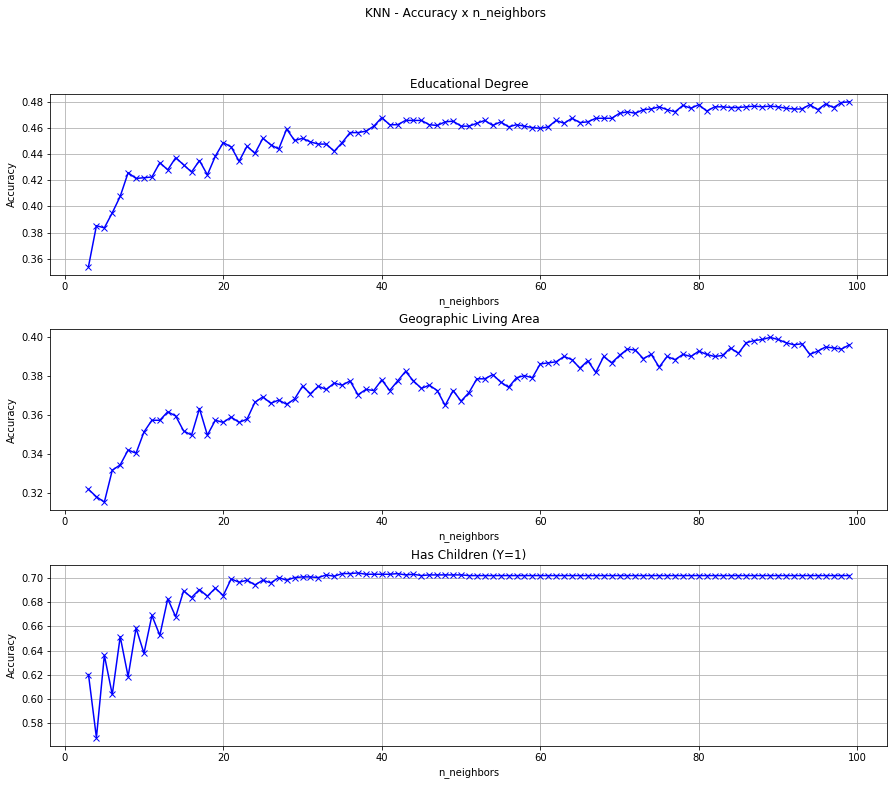
A picture containing indoor

Description automatically generated

*Figure 2 - Features histograms after removing outliers*

# Categorical features classification

Initially, the Educational Degree needed to be encoded for numerical values with LabelEnconder from scikit learn. The numerical features were then rescaled to be within the [0,1] range using the MinMaxScaler. Later, the k-nearest neighbor algorithm KneighborsClassifier was applied and its accuracy for each feature was evaluated with a custom function which computed the confusion matrix for each trained model. The results are shown in Figure 3.



*Figure 3 - Classifier accuracy for each categorical feature*

Since we didn't achieve good estimations for the first two categorical features, the team decided to state the minimum accuracy at 2/3 and to drop the rows which contained NaNs on these two columns. Next, the optimal value for the number of neighbor samples (k) to Has Children (Y=1) was set to 21.

**k choice for the kNN algorithm:** the ‘rule of thumb’ is to choose k = sqrt(n)where n stands for the number of samples in the training dataset and k is the number of instances that we take into account to determine affinity with classes. Since we chose 80% of the data to be the training dataset, k would be around 89. The REF \_Ref27327102 \h Figure 3 instead shows that there would be no improvement in selecting a number higher than 21 for Has Children (Y=1).

# Numerical features regression

The first approach to fill the Numerical NaNs was to create a function that train 3 different regressors, split into complete (rows without NaNs) and incomplete (rows with at least 1 NaN) and then check the R Squared Error (R2E) of each algorithm. We’re going to use the one with the best R2E. The algorithms tested are:

* DecisionTreeRegressor
* LinearRegressor
* LinearSVR

Using this technique only ensured that we would have the smallest MSE out of these 3 algorithms, but it was not enough to achieve good results. The parameterization of the correct algorithm would be crucial for the result to improve. For this, DecisionTreeRegressor was selected as an algorithm and a Genetic Algorithm (GA) implementation was made to try to achieve a good R2E. We will not delve into the explanation of the generic algorithm, but we made a unique report for it and made it available along with the code in the GitHub link that will be at the end of this report.

The group decided to use the DecisionTreeClassifier with the result of the GA only in the columns that it achieves more than 0.60 R2E and the rest will be dropped.

The result of the Genetic Algorithm showed that only in the 'Premiums in LOB: Motor' and 'Premiums in LOB: Life' columns the results were above 0.60, being 0.8157 and 0.6286 respectively. The NaN values ​​of the cited columns were filled according to the parameterization selected by the GA.

# Drop the rest of NaNs

After applying the regressor we still had 206 NaNs. We dropped the rest of them.

# Data preprocessing

This chapter presents the techniques applied to analyze the data and the findings that emerged from them.

# Features correlation

Plotting the correlation matrix for all features (Figure 4) we can see that Age is highly correlated to Gross Monthly Salary. Since Age is also an untrustworthy feature (see section 2.2) and the amount of information would remain in the correlated feature, the team decided to drop the Age column for the analysis.

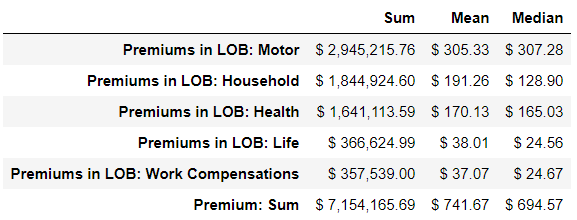
The Claims Rate also has a high inverse correlation with Customer Monetary Value (CMV), which makes sense since the Claims Rate is related to the profit that the company makes with each customer. So, the team decided to again reduce the input space and later clusterize the dataset only on the CMV perspective, since it carries more information than Claims Rate (includes customer retention and acquisition cost).

A screenshot of a cell phone

Description automatically generated

*Figure 4 - Correlation matrix and heatmap for all features*

What also calls our attention is that Premiums in LOB: Motor Insurance is considerably correlated (inversely) with all other insurance premium totals. Considering that signing a policy in any group could lead to increase proneness to contract other kinds of insurance (complementary products), this behavior is unexpected. Taking into account that the mean premium for Motor is substantially higher than for the other groups (Figure 5), it's contracting may be affecting the capacity or interest in paying for the other products. A possible insight would be to focus more on reverting this tendency.



*Figure 5 - Premiums measures*

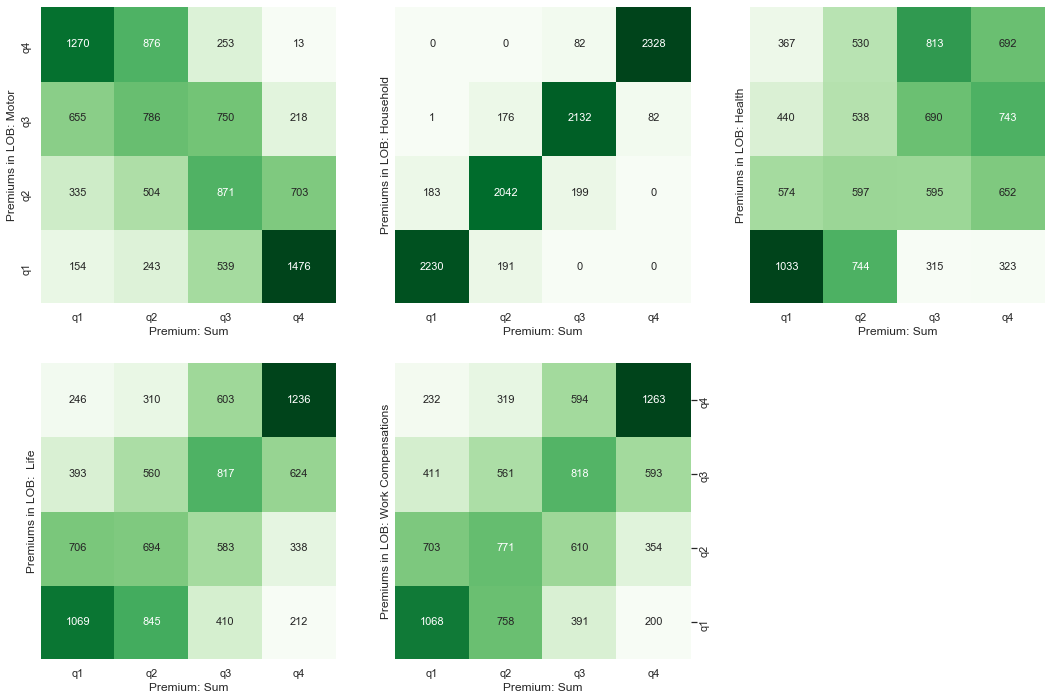
Premiums in LOB: Household, Premiums in LOB: Life and Premiums in LOB: Work Compensations correlation show that these groups are decently working as complementary products, and its correlation could also be improved.

Finally, it’s possible to note a perfect correlation between the amount in Household insurance and the sum of the premiums, a fact that can also be commercially explored by the company.

# Analysis

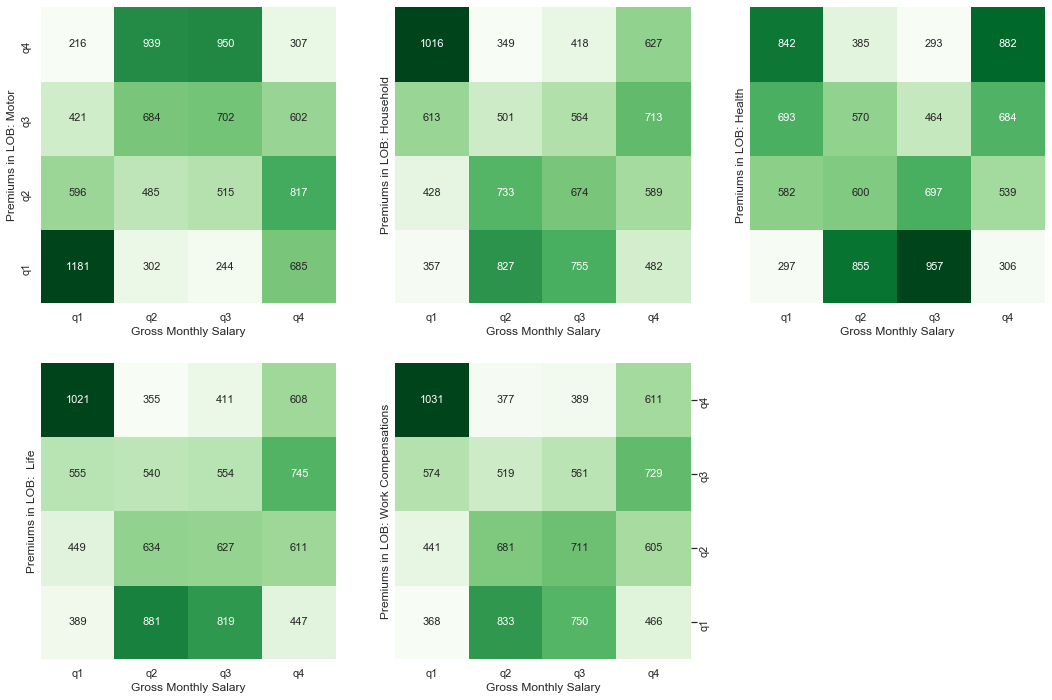
# Quartiles (a priori grouping)

Separating the premiums data into quartiles confirms the inverse correlation between Motor insurance consumption and the sum of premiums (Figure 6). One good strategy would be to focus on inverting the correlation. Regarding the other insurance groups, specifically Health, Life and Work Compensations, they show a scattered but still correlated distribution over the quartiles. Possible approaches for these findings would be to take actions in order to bring customers closer to the main diagonal, i.e., increase the premiums total for customers that already have high premiums in any of these groups, and increase consumption on specific groups when the customer has already a high premiums total.



*Figure 6 - Quartiles plot for insurance groups vs premium sum*

Another perspective that was also analyzed by the team was the premium amount for each group versus the gross monthly salary (Figure 7), which, in principle, could reveal the customer potential for contracting insurance. Ther quartiles show that this potential is not being totally tapped. Stands out, for instance, the customers' group in Q3 for Gross Monthly Salary and Q1 for Premiums in LOB: Health.

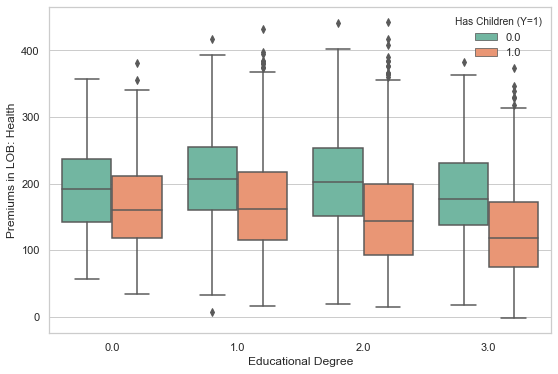


*Figure 7 - Quartiles plot for insurance groups vs gross monthly salary*

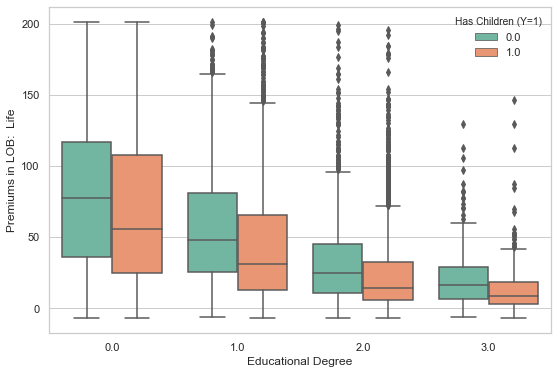
# Boxplots

A further investigation continued with boxplots. The team was particularly interested in understanding the changes in the distribution of the insurance consumption features under the different customer categorical features.

Considering that people with children and higher education levels would be more prone to pay higher premiums in Health and Life, since the premium is proportional to the insured sum and the coverages, there is an unexpected behavior on the plots of Figure 8 and Figure 9. The same occurs with Work Compensations and Household, and the opposite is observed for Motor insurance. It seems that clients with high insurance buying potential are not being accessed for events that can cause major changes in their lives.



*Figure 8 – Health premiums distribution on the educational degree and children in the family.*



*Figure 9 – Life premiums distribution on the educational degree and children in the family.*

# Decision tree

Trying to predict the Premium: Sum or the Customer Monetary Value quartiles of a customer from it’s profile (Educational Degree, Geography, Children, Salary and First Policy´s Age), resulted in a maximum accuracy of only 0.357. On the other hand, the graph confirms the behavior observed so far and reveals that the geography is not a relevant variable for customer potential measurement (Figure 10).

A close up of a piece of paper

Description automatically generated

*Figure 10 - Decision tree for quartiles of Premium: Sum*

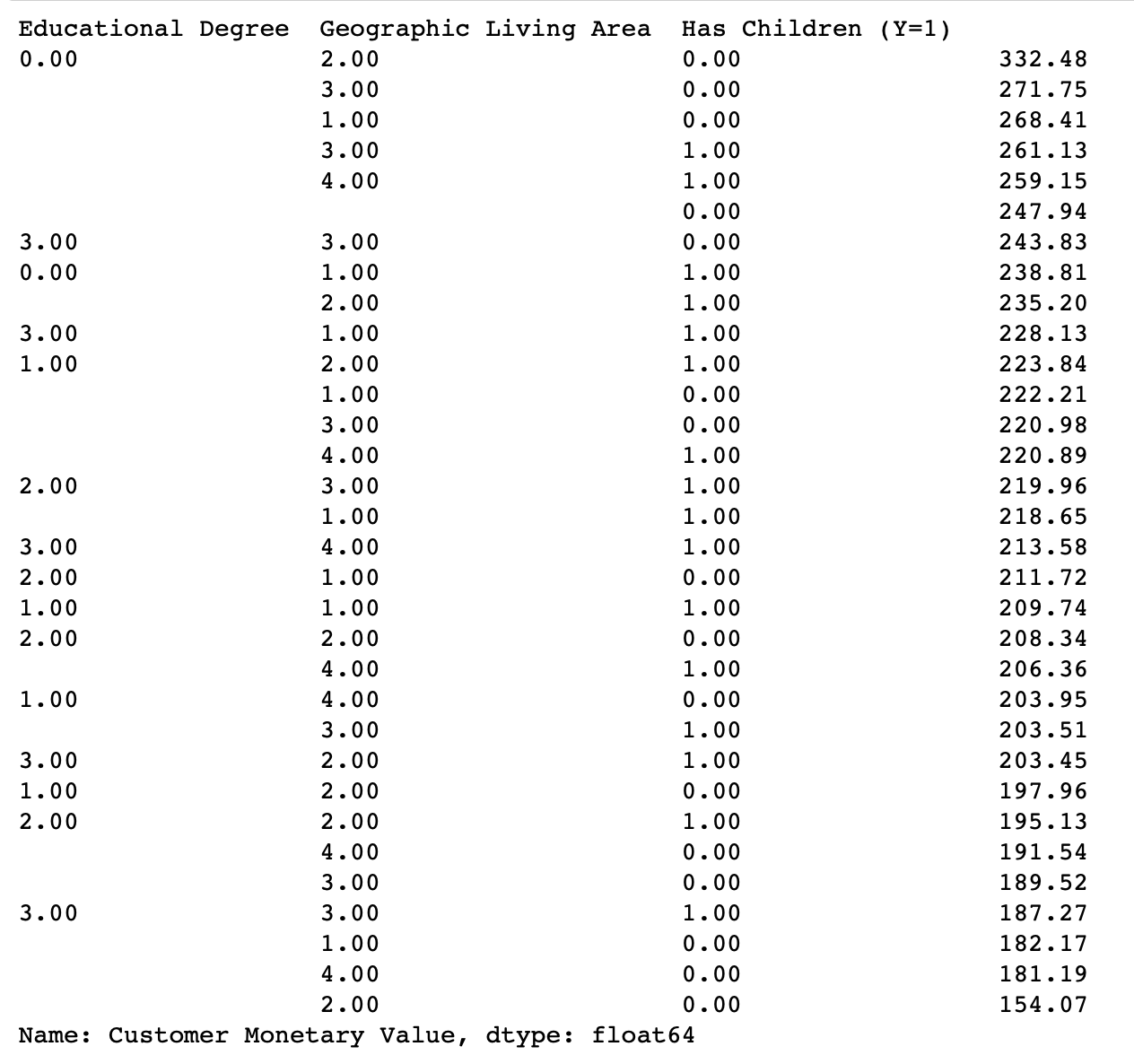
# Clustering

## Groupby Table

After removing all of the outlying data, we can try to build clusters in order to better understand our customers. As categorical and continuous data cannot be easily used together in a clustering algorithm, we can begin by looking at just the categorical data. We can use a groupby to separate the data into unique groups (similar to the RFM method).

As “Customer Monetary Value” is a good target variable (it shows a customer’s value to the company), we can aggregate it with either the mean or median and then group it by each categorical data. We can then sort each group by its “Customer Monetary Value” average such that we see which set of categorical parameters are on average better or worse for the company

As the dataset no longer includes any significant outliers, we should expect that the median and mean should be nearly the same value. However, we see that only the mean returns a meaningful result for “Has Children” and “Educational Degree”. Customers with no children have a slightly higher average claims rate than customers with children. The six groups with the highest “CMV” average all have a basic education. The rest of the groups are randomly spread across the different education levels. This suggests that people with the lowest level of education have the highest value to the company, while other levels of education have not only a lower level of value to the company, but also cannot be differentiated from each other.

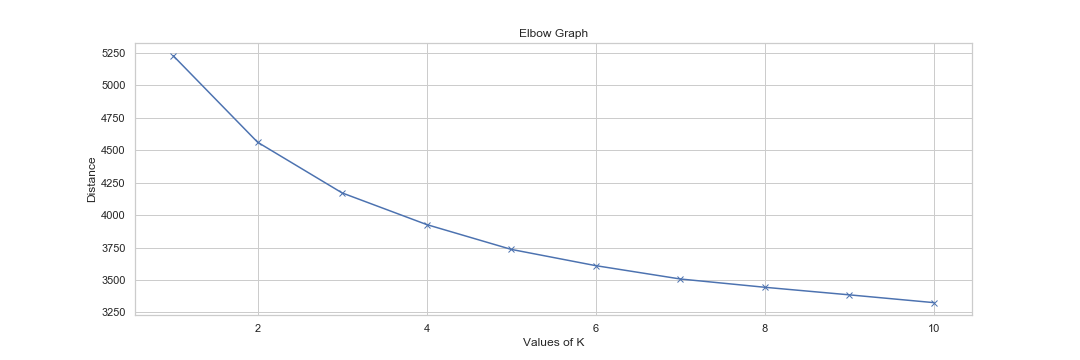


While these observations disappear when using the median as the average, we see that the averages have a larger range. This is most likely a result of the mean averaging the people with a ratio of 0 and the median taking the 50th percentile of customers in that group.

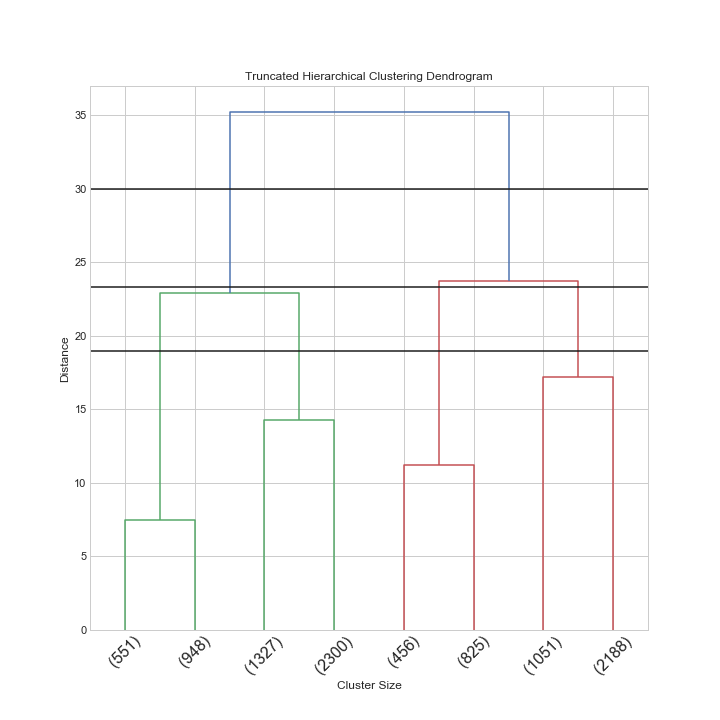
Using count instead of the average also reveals that half of our customers have children, live in area 1 or 4 and have an education of 1 or 2. Using this data can help to target advertising towards potential customers. This gives us 4 groups to target instead of the 32 possible groups.

## K-Means

We can use the k-means algorithm to cluster the numerical data. In order to do this, we must define the number of clusters. For this we can run the k-means algorithm several times and build an elbow plot. We can also build a dendrogram to confirm the results in the elbow plot.



From the elbow plot in figure (), we can see that a significant elbow does not really appear, but a slight one appears at either 2, 3 or 4 clusters. We can create a dendrogram to compare.



The black horizontal lines are drawn such that the cut the dendrogram into 2, 3 and 4 clusters. Looking at the number of customers in each branch (indicated by the number at the bottom), we can see that using 3 would create a cluster of around 1000 customers while the other clusters are much larger. 4 clusters seem appropriate as using 4 reduces the distance significantly over 2 while also keeping the number of clusters similarly sized.

Running the k-means with 4 clusters does not give a satisfactory result when plotting any two random variables. If we look at figure (), we can see that the clusters have no meaning as they simply break the data into two equal parts (2 clusters would be equivalent here). Each graph is plotted with the “gross monthly salary” as the x-axis. Plotting with a different x-axis (motor insurance premiums) in figure () returns more interesting shapes (triangles rather than the circles or squares here). As the triangles show a negative correlation, we can predict that customers who spend a lot on motor insurance spend less on other insurances. All of the 2 variable plots are presented in the appendix.

## Mean Shift

Using mean shift is in theory a good choice as it will allow us to find clusters without specifying the number of clusters beforehand. However, there can be a problem as we need to specify the bandwidth used in the mean shift algorithm. If we do not specify it, the algorithm can estimate it. For this dataset the algorithm finds 3 clusters. If we change the bandwidth manually we can “force” it to have as many clusters as we want (by lowering the value of bandwidth), but this of course defeats the point of this algorithm. The bandwidth parameter is also very sensitive as lowering it too much means that many of the further points become outliers and the number of clusters increases unreasonably.

Unlike k-means, the clusters are more or less clearly defined for most variables when plotted against each other. The algorithm does suffer from lots of noisy points that make defining the border between clusters difficult. The plots are presented in the appendix

## Gaussian Mix

Running the gaussian mix algorithm requires the number of clusters to be specified. If we specify 4 clusters, we do not get a significant fourth cluster. This means that the algorithm is finding 3 real clusters. Gaussian Mix is better than k-means and has more defined borders than Mean Shift but one of the clusters is significantly larger than the other two. Mean shift gives more equally sized clusters.

## DBSCAN

While DBSCAN does not require the number of clusters to be set before running, it does require setting two parameters: the radius of the selection circle and the number of points in the aforementioned circle for that point to be used to select new points for the cluster. As the data in this set are very close together, DBSCAN will not work very well and the results confirm this. With most values for the parameters for this dataset, the algorithm will return one cluster and if the parameters are changed such that a second or third cluster is produced they are very small and are not of any significant size. With more clusters, the algorithm breaks down and just returns lots of noisy points while maintaining the larger central cluster.

## Kmodes

Using k-modes is a good alternative to k-means for categorical data but only when the categorical data contains lots of different values. In this case, we only have two or four unique values for each attribute. Because of this as well as needing to pick the number of clusters does not make k-modes a very useful algorithm for this dataset. Even using two clusters is worse than the groupby tables used previously.

# Summary