# **To-do:**

* Explain, why AgglomerativeClustering wasn’t used
* Include our environment in the hand in

# **Project Report for**

# **Data Mining Group Project**

# 

## **Master’s in Data Science and Advanced Analytics at NOVA IMS, Lisbon**

## **Group Information**

Group H

Group members:

Henrique Vaz

Philipp Metzger

Link to GitHub repository:

<https://github.com/ph1001/NOVA-Data-Mining-Project>

**Abstract**

150-250 words, one paragraph.

Summarise: Introduction, Methodology, Results and Conclusions

**I. Introduction**

The goal of this project is to develop a customer segmentation based on a dataset provided by the non-profit organization Paralyzed Veterans of America (PVA). PVA provides programs and services for US veterans with spinal cord injuries or disease. The dataset contains information on individuals that have donated to PVA and that are classified as “Lapsed” donors[[1]](#footnote-1), meaning that they made their last donation to PVA 13 to 24 months ago.

**II. Background**

In chapter 1 of [1], the goal of “clustering” is described as the discovery of groups of similar examples within the data. The customer segmentation in this project was done by clustering the observations contained in the dataset provided by PVA, resulting in subgroups of similar observations, to which then through interpretation of the characteristics of the different clusters, different marketing approaches were assigned.

In chapter 9.1 of [1], the intuition of clustering is described as the effort to find subgroups in the dataset at hand, such that the inter-point distances between points of the same cluster are small in comparison to their distances to points from different clusters.

There are several clustering techniques. One of the most popular ones is the k-means clustering algorithm. The goal of k-means is to partition the data into k subgroups, where every datapoint is allocated to one and only one subgroup. The process of k-means clustering can be summarised in the following way: k vectors from the same vector space as the points from the dataset are chosen in an appropriate manner. These k vectors are called centroids. For all points in the dataset, the nearest centroid is identified, using a distance measure such as the Euclidian distance, and the point is assigned to it. Then, all k centroids are updated in such a way that their new position represents the centre of the subgroup of points that were assigned to each one of them. Then, the assignment of every point to the nearest centroid is repeated and after that, the position of the k centroids is updated accordingly. These steps are repeated until a stopping criterion is fulfilled. The resulting k centroids represent the centres of k clusters in the dataset, which are comprised of the points that are nearer to the respective centroid than to any other one.

**III. Methodology**

**III.1 Materials and Software**

The materials used in this project are the dataset provided by PVA and the related metadata file, which describes that variables contained in the dataset. The dataset consists of 95412 observations and 475 variables.

The software that is used in order to complete the project is Python and more precisely Anaconda and Jupyter Notebook. In the latter, the code for this project is created.

In the following part, the steps conducted in our Jupyter notebook are described. The order of these descriptions is the same as the order of actions undertaken in the Jupyter notebook that has been handed in alongside with this report. The results and findings as well as their implications will be presented and discussed in the following chapter IV.

**III.1 Imports and Organisation of Libraries and Data**

As a first step, the necessary libraries were imported, and the dataset was loaded into main memory from the csv file provided.

Next, using the file “pva\_metadata.txt”, the features obtained from the imported dataset were split into metric and non-metric features.

**III.2 Data Cleaning**

In this step, again using the metadata file, the existence of cells was assessed, where spaces (“ “) carry a meaning, such as for the feature ‘MAILCODE’, where a space means that the address of this individual “is OK”, whereas the value “B” means that the address “is bad”. Other features where something like this is the case are the features ‘NOEXCH’, 'RECINHSE', 'RECP3', 'RECPGVG', 'RECSWEEP' and ‘MAJOR’. For these features, the spaces were replaced by a meaningful string, such as “Address is OK”.

In the next step, the existence of duplicated observation was assessed and all remaining spaces in the dataset were replaced by NaN values[[2]](#footnote-2). It was also checked if the dataset contains any empty strings.

In the next part of the code, the percentage of missing values contained in each feature was assessed. Features that have more then 40 % missing values were discarded.

**III.3 Data Transformation**

In this section of the Jupyter notebook, all features containing time related information such as dates were transformed from their original string format to the datatype datetime.date[[3]](#footnote-3). For this purpose, using the metadata file, a list ‘date\_features’ was defined, which contains all the features that are to be changed in this step. Each column represented by an element of this list, is then sent through a pipeline consisting of three functions that were defined in this step. Part of the functionality of this pipeline is to ensure that NaN values remain unchanged.

**III.4 Further Data Cleaning**

In this section of the code, the distributions of some features, where anomalies had caught the authors’ eyes, were first visualised in order to then remove the unusual patterns that are likely to be errors stemming from a faulty process of data collection. Then, the unusual values were replaced by NaN values.

**III.5 Feature Selection**

As a first feature selection step, correlations between all metric features were assessed. Of feature pairs that were highly correlated, one was discarded, and one was kept in the dataset. Also, metric features that only contain a very small number of distinct values and thus carry little information for our analysis as well as features containing mostly zeros were discarded.

**III.6 Further Feature Transformation**

In this step, the date features whose transformation was described in III.3 were further transformed to integers representing their distance in days to the reference date stored in each observation’s value of the variable ‘ADATE\_2’[[4]](#footnote-4). For better traceability, the resulting features were renamed in the following format: “<<original feature name>>>\_rel\_in\_days”.

**III.7 Perspectives Creation**

At this point of the project there are more than 200 remaining features, which represent very high dimensionality. In order to reduce this dimensionality but still keep most of the information, the data was divided into perspectives.

Using only our metric features and the respective description, groups of similar features were created. The idea is to have different sets of features that contain the same type of information.

Missing steps?

**III.8 Clustering**

For the clustering, two approaches were followed:

1. The basic idea of this first approach is to perform clustering on a dataset filled with clustering labels. The latter generated by clustering each perspective as a single dataset.

This process can easily be defined and better understood analysing it step by step:

1. Define which perspective are going to be used.
2. Apply different clustering solutions to each perspective and keep a suitable one.

After getting the right solution, add the labels to each point clustered.

Following that save the characterizations of those clusters on a data frame that contains the perspectives’ columns grouped by label using the mean value of each label (getting the centroid of each cluster).

1. Concatenate in a new data frame all the labels obtained in the different perspectives clustered. This results in a *n* x *p* dataset, where *n* represents the number of rows in the perspectives (that is the same for all) and *p* represents the number of perspectives used.
2. After finding this *n* x *p* dataset the final phase of this approach is reached. In this step the user should find a suitable clustering algorithm to apply on this dataset. Having this result, the goal is to use the characterizations mentioned in step 2 to characterize our final labels.

The reason for this approach to be taken is related with the high dimensionality of the data. A way to work around that issue is to try to break data into different parts and apply clustering separately.

1. K-means clustering on different feature sets, representing different perspectives on the data with subsequent combination of those perspectives and hierarchical clustering on the found centroids in order to determine the perspective combination that leads to the highest R2 score.

In the following, the second approach will be described in detail:

First, the feature sets representing the different perspectives on the data were assessed again, drastically reducing their number and their number of features per perspective. For this, the component planes from approach 1 were used to identify the most promising features from the features used in approach 1. The component planes for the perspective ‘donor\_info’ are presented in Figure 1. From these four features, three were selected to be used in this approach: ‘ODATEDW\_rel\_in\_days’, ‘INCOME’ and ‘NUMPROM’.

A picture containing graphical user interface

Description automatically generated

Figure 1: Component planes that were created in clustering approach 1

Then, on each of these reduced feature sets, k-means clustering was done iteratively with values of k ranging from 2 to to 9, each time saving the R2 score of the resulting clustering solution. Like this, the optimal value for k was identified for each perspective. Figure 2 shows the resulting R2 scores for each perspective and for varying values for k.

Chart, line chart

Description automatically generated

Figure 2: R2 scores for the different perspectives and varying values of k (numbers of clusters)

Then, with the optimal value for k for each perspective, k-means was run again once more on each perspective, yielding cluster labels for each perspective. In the next step, pairwise combinations of all perspectives were created and the centroids for each of the the resulting cluster label pairs were computed. Each of the resulting arrays of centroids were then the basis for hierarchical clustering, with the purpose of assessing which of the cluster pair combinations could be combined into merged, larger clusters. For this ‘AgglommerativeClustering’ was used on each result of each pair of perspectives and then a dendrogram for each fitted instance was created, in order to visually assess, which number of cluster is appropriate. These values were manually saved in a list and used for another hierachical clustering step, in which it was determined, which pair of perspectives yields the best R2 score. After determining this, hierarchical clustering was re-run with this pair of perspectives and and the optimal number of clusters obtained from the visual inspection of the dendrograms described before. A mapper was used to apply the result to the whole dataset.

**IV. Results** **and Discussion**

**IV.1 Results: Imports and Organisation of Libraries and Data**

We identified 398 metric and 77 non-metric features in the dataset.

**IV.2 Results: Data Cleaning**

In this step it was found out, that no duplicated observations or empty strings exist in the dataset. Furthermore, 3011889 spaces were converted to NaN values.

67 metric and 30 non-metric features were discarded from the dataset, due to them having a percentage of missing values higher than 40 %.

**IV.3 Results: Data Transformation**

The result of this step is the updated dataset, where the features defined in the list ‘date\_features’ have been changed to objects of the type datetime.date. This facilitated their further processing and enabled us to do calculations such as addition or subtraction of different date columns.

**IV.4 Results: Further Data Cleaning**

The variables with the names ‘AGE90x’, x ∈ {1, 2, 3, 4, 5, 6, 7} serve as a good example for the process of removing values that are likely to be faulty. Figure 1 shows the distributions of these features before any values were removed. It is apparent, that there are values similar to zero, whose frequencies don’t integrate well with the rest of the frequencies of these distributions. These values are indeed the value zero. After all of these zeros were replaces by NaN, the distributions were checked again. The resulting distributions, visualised as histograms, are presented in Figure 2.

Graphical user interface, diagram

Description automatically generated

Figure 3: The distributions of the variables ‘AGE90x’, x ∈ {1, 2, 3, 4, 5, 6, 7} before the removal of the value zero

**A picture containing chart

Description automatically generated**

Figure 4: The distributions of the variables ‘AGE90x’, x ∈ {1, 2, 3, 4, 5, 6, 7} after the removal of the value zero

**IV.5 Results: Feature Selection**

In this step, 92 features were discarded. The number of metric features was decreased from 331 to 249 and the number of non-metric features dropped from 47 to 37.

**IV.6 Results: Further Feature Transformation**

The values in days resulting from this step range from -122 days (= approximately -0.33 years) to 31928 days (= approximately 87 years). After the discovery that negative values exist, they were located in the dataset. They belonged to the feature ‘DOB’, or more precisely to the newly created feature ‘DOB\_rel\_in\_days’. There were five observations present in the dataset that had values equal to or smaller than zero in this column[[5]](#footnote-5). It was decided to remove these observations, since it is highly unlikely that any promotion was mailed to an individual that wasn’t born yet or that was born on the day of the mailing.

**IV.7 Results: Perspectives Creation**

As stated in III.7, different sets of features were generated. Each set of features represents a perspective/category. The result of this process is a total of 25 different perspectives, which can further be used for clustering.

The decision of using or not a specific perspective will depend not only on the average value of GINI Index of the perspective’s features, but also on some intuition of what the authors consider is relevant for the result. This decision process led to a reduction from 25 to 15 perspectives to be clustered.

The perspectives that went on to be clustered further ahead were the following:

1. Donor Info: personal information about the donor.
2. Donor Third-Party: information of third-party entities regarding the donor.
3. Donor Gifts Given: information of gifts given by a donor.
4. NB1 Population: information about population on the donor’s nb.
5. NB Provenance: information about provenance of a donor’s nb.
6. NB Ages: information about donor’s nb population aging.
7. NB Housing Price: information about house pricing in donor’s nb.
8. NB House Quantity: information about number of house of donor’s nb people.
9. NB People Held: information about amount of people held in house in the nb.
10. NB Income: information about the incomes on donor’s nb.
11. NB House Moving: information about house changing in donor’s nb.
12. NB Life Occupation: information about donor’s nb’s people occupations.
13. NB Job: information about donor’s nb’s people jobs.
14. NB Education: information about donor’s nb’s people education level.
15. NB Military Service: information about presence of military service in donor’s nb.

**IV.7 Results: Clustering**

It is important for the reader to keep in mind that this approach follows a single procedure and applies it to the different existing perspective. In the end it stores the results of each perspective’s clustering solution in a final data frame composed only by labels.

Hierarchical Clustering on top of KMeans:

In this approach the first option to be taken was to apply hierarchical clustering on top of kmeans. For the kmeans algorithm the *n\_clusters* chosen was 1000. This way we would end we 1000 centroids and further on apply hierarchical clustering on those values.

After running kmeans several Nan values appeared. In addition to this, the application of hierarchical clustering on those values wasn’t either performant or suitable. Thus, this solution was discarded.

KMeans on top of SOMs:

In a second attempt, that intended to follow the same logic but with different techniques, KMeans was tested on the result of different *Self Organizing Maps* (SOM)results (units).

For the first and unique SOM created the chosen map size was 50x50, thus resulting in 2500 units as expected. This huge number of units allow the user to have a good amount of data to cluster in further actions. Other relevant observation to point out is that the same number of epochs was chosen for the unfolding and finetuning phases, 100. The value represents a balance of good performance in short time and a good minimization of the quantization error.

Right after getting the result of SOM, KMeans is applied on top of that result. The values of *n\_clusters* for KMeans tested ranged in the interval [3, 7]. Although the results were not stored, the authors still found out that the best results were shown with *n\_clusters* = 6 based on the different r2 obtained. Adding to the fact that the r2 were higher, keeping a higher number of clusters in this phase will ensure more variance in the final labels’ dataset.

After the kmeans is ran, the labels are added to each perspective’s data frame resulting of a data frame that has the perspective’s columns plus a column with a label assigned to each observation.

It’s also important to store each perspective clustering description. For this, n\_prespective data frames were created.

Table

Description automatically generated

Table 1: Characterization of Donor Info perspective clustering solution

Table

Description automatically generated

Table 2: Characterization of Gifts Given perspective clustering solution

Table

Description automatically generated

Table 3: Characterization of NB Population perspective clustering solution

A picture containing table

Description automatically generated

Table 4: Characterization of NB House Price perspective clustering solution

Table

Description automatically generated

Table 5: Characterization of NB Income perspective clustering solution

Table

Description automatically generated

Table 6: Characterization of NB Job perspective clustering solution

Table

Description automatically generated

Table 7: Characterization of NB Education perspective clustering solution

Table

Description automatically generated

Table 8: Characterization of NB Military Service perspective clustering solution

Whenever a single perspective has been put through SOM and KMeans, its resulting labels’ column is added to the final data frame. This final data frame is described in III.8 a) step 3.

Having obtained the final data frame (Table 9), filled only with labels and having the same number of columns as number of perspectives, different clustering algorithms were directly applied on it.

Table

Description automatically generated

Table 9: Final data frame with labels only

The first one that was tried was a KMeans. The number of clusters chosen to use in this phase was based on an inertia plot, using the elbow method.

Chart, line chart

Description automatically generated

From the plot above, the number of clusters chosen was 5. This number is right after the great decay in SSW and represents a decent number of clusters to describe in the end.

Using *n\_clusters* = 5 the obtained r2 was 0.32, a value that’s higher the ones in the tries with *n\_clusters* < 5.

Besides Kmeans, other algorithms were tested. One of these other was Meanshift clustering. To initialize this algorithm the bandwidth was estimated, having as a result of that a value around 8. Using that bandwidth value, the result of the algorithm retrieved only one single cluster, which is not suitable.

Moreover, DBSCAN was also ran over the final data frame (Table 9). To find the optimal eps value to use in DBSCAN a k-distance graph was plotted, with parameter *n\_neighbors* = 300 that is roughly the square-root of the number of observation of the data.

Chart

Description automatically generated

The value that was picked was 3. After that value the eps starts increasing too much and would not suit the purpose of the application. However, even picking the right value for the eps, the number of clusters retrieved with DBSCAN was also 1. Once again, this is not a solution that would fit the needs of this project.

Finally, the last attempt to cluster the final data frame (Table 9), was using hierarchical clustering. The type of linkage chosen was the only one available throughout all the process, single. Using that linkage, the r2 scores were retrieved, from different numbers of clusters combinations. The results for 2 to 9 clusters were all bellow 0.0001. Thus, this solution also ends up discarded.

To conclude this final clustering phase, the chosen model was the KMeans and those are the results to take into consideration.

Next steps

**V. Conclusions**

**VI. References**

|  |  |
| --- | --- |
| [1] | C. Bishop, Pattern recognition and machine learning, Springer, 2006. |
| [2] | [Online]. Available: https://docs.python.org/3/library/datetime.html. [Accessed 30 Dec. 2020]. |

1. In the dataset’s metadata file also denoted as “Lapsing” donor [↑](#footnote-ref-1)
2. NaN, short for “not a number”, commonly denotes a missing value in a dataset. [↑](#footnote-ref-2)
3. For the documentation of the Python library “datetime”, see [2] [↑](#footnote-ref-3)
4. ‘ADATE\_2’ represents the dates on which the most recent promotion was sent to each individual. This variable was chosen to be the reference time for each observation. [↑](#footnote-ref-4)
5. These observations have the following indices: 22984, 40565, 52252, 60753, and 94452. [↑](#footnote-ref-5)