# **To-do:**

Explain, why AgglomerativeClustering wasn’t used.

# **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 provide 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 is 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 are assigned.

In chapter 9.1 of [1], the intuition of clustering is described as the effort to finding 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 could be summarised in the following way: k vectors from the same 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 are imported, and the dataset is loaded into main memory from the csv file provided.

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

**III.2 Data Cleaning**

In this step, again using the metadata file, the existence of cells is 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 or similar is the case are the features ‘NOEXCH’, 'RECINHSE', 'RECP3', 'RECPGVG', 'RECSWEEP' and ‘MAJOR’. For these features the spaces are replaced by a meaningful string, such as “Address is OK”.

In the next step, the existence of duplicated observation is assessed and all remaining spaces in the dataset are replaced by NaN values[[2]](#footnote-2). It is 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 is assessed. Features that have more then 40 % missing values are discarded.

**III.3 Data Transformation**

In this section of the Jupyter notebook, all features containing time related information such as dates are 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’ is 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 are defined in this step. Part of the functionality of this pipeline is to ensure that NaN values remain NaN values.

**III.4 Further Data Cleaning**

In this section of the code, the distributions of some features, where anomalies had caught the autors’ eyes, are 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 are replaced by NaN values.

**III.5 Feature Selection**

As a first feature selection step, correlations between all metric features are assessed. Of feature pairs that are highly correlated, one is discarded, and one is 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 are discarded.

**III.6 Further Feature Transformation**

In this step, the date features whose transformation was described in III.3 are 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 are named in the following format: “<<original feature name>>>\_rel\_in\_days”.

Next steps

**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 are 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 are replaces by NaN, the distributions are checked again. The resulting distributions are presented in Figure 2.

Graphical user interface, diagram

Description automatically generated

Figure 1: 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 2: 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 are discarded. The number of metric features is decreased from 331 to 249 and the number of non-metric features drops 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 belong to the feature ‘DOB’, or more precisely to ‘DOB\_rel\_in\_days’. There were five observations present in the dataset that had values equal or smaller than zero in this variable[[5]](#footnote-5). It was decided to remove these observations, since it is not possible or at least 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.

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 indices of 22984, 40565, 52252, 60753, and 94452. [↑](#footnote-ref-5)