# Capstone Project: Customer Segmentation Report for Arvato Financial Services

Desared Osmanllari

May 23, 2020

## 1 Problem Statement

In the financial world, most of the decisions are becoming more and more data-driven. Arvato Financial Solutions offers all financial services related to payments and cash flow - from risk assessment to the emergence of receivables through invoicing and settlement[1]. Arvato is analysing various datasets and attributes to calculate credit scores, detecting fraud by analysing the payment behaviour, and finding the proper customers in a given population. These problems can be solved by applying machine learning.

In this project, supervised and unsupervised learning techniques will be used to analyze demographics data for customers of a mail-order company (Arvato Financial Solutions) in Germany against demographics information of the existing clients. Therefore, the research question that arises is:
What are the potential customers and how can the company acquire them efficiently?
The main purpose of this project is to identify the right people, who can be potential future customers. It will increase the efficiency of the customer acquisition process, by targeting the proper clients. On the other hand, without a data driven approach, the company would advertise their product to the whole population in Germany, which can be both inefficient and costly.

The project is designed in four main sections.

**1. Data Preprocessing:** Data preparation, cleaning, and transformation occurs in this section. Feature engineering is further required. These

[1]https://finance.arvato.com/en/

are very important steps which help building an efficient machine learning model.

**2. Customer Segmentation:** In this part, unsupervised learning techniques will be used to perform customer segmentation. Principal component analysis (PCA) will be used for dimensionality reduction. Then, the elbow curve will be used to identify the most optimal number of clusters fitting the KMeans algorithm. Finally, KMeans will help the segmentation of population and will determine which of these segments is more similar to the real customers.

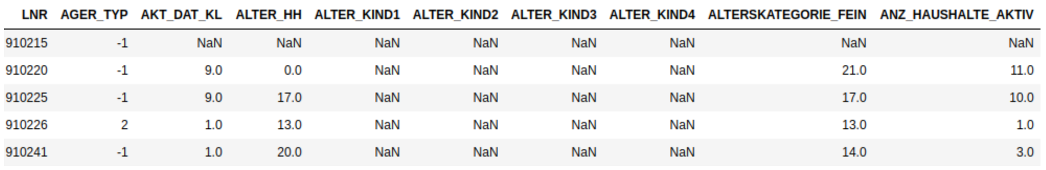
**3. Supervised Learning Model:** A machine learning model will be trained using historical responses of marketing campaigns. This model will be further used to predict which individuals are most likely to convert into becoming customers for the company. Several machine learning algorithms will be used to build the model, while Grid Search will be used to tune the hyper-parameters. ROC-AUC curve will be used to assess the model performance.

**4. Kaggle Competition:** Last step is to use the most efficient model to make predictions in a test set. The results will be submitted in the Kaggle competition.

## 2 Datasets and Inputs

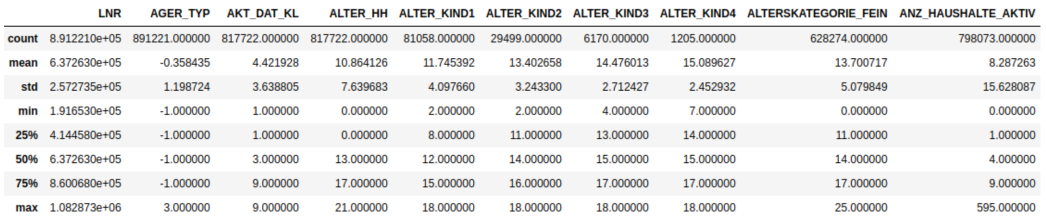
The data is provided by Bertelsmann Arvato Analytics, and it includes the general population dataset, the customer segment dataset, the mailout campaign dataset and a test dataset.

*Udacity_AZDIAS_052018.csv*: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns). A sample of data is shown in figure 1. Some of the feature values are missing. The data types are categorical, binary, and numerical. Descriptive statistics of several attributes are shown in figure 2.



| LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTERSKATEGORIE_FEIN | ANZ_HAUSHALTE_AKTIV |
|---|---|---|---|---|---|---|---|---|---|
| 910215 | -1 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| 910220 | -1 | 9.0 | 0.0 | NaN | NaN | NaN | NaN | 21.0 | 11.0 |
| 910225 | -1 | 9.0 | 17.0 | NaN | NaN | NaN | NaN | 17.0 | 10.0 |
| 910226 | 2 | 1.0 | 13.0 | NaN | NaN | NaN | NaN | 13.0 | 1.0 |
| 910241 | -1 | 1.0 | 20.0 | NaN | NaN | NaN | NaN | 14.0 | 3.0 |

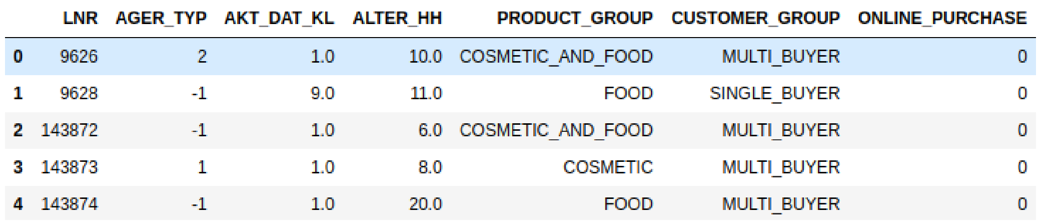
Figure 1: Sample data of general population



| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTERSKATEGORIE_FEIN | ANZ_HAUSHALTE_AKTIV |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 8.912210e+05 | 891221.000000 | 817722.000000 | 817722.000000 | 81058.000000 | 29499.000000 | 6170.000000 | 1205.000000 | 628274.000000 | 798073.000000 |
| **mean** | 6.372630e+05 | -0.358435 | 4.421928 | 10.864126 | 11.745392 | 13.402658 | 14.476013 | 15.089627 | 13.700717 | 8.287263 |
| **std** | 2.572735e+05 | 1.198724 | 3.638805 | 7.639683 | 4.097660 | 3.243300 | 2.712427 | 2.452932 | 5.079849 | 15.628087 |
| **min** | 1.916530e+05 | -1.000000 | 1.000000 | 0.000000 | 2.000000 | 2.000000 | 4.000000 | 7.000000 | 0.000000 | 0.000000 |
| **25%** | 4.144580e+05 | -1.000000 | 1.000000 | 0.000000 | 8.000000 | 11.000000 | 13.000000 | 14.000000 | 11.000000 | 1.000000 |
| **50%** | 6.372630e+05 | -1.000000 | 3.000000 | 13.000000 | 12.000000 | 14.000000 | 15.000000 | 15.000000 | 14.000000 | 4.000000 |
| **75%** | 8.600680e+05 | -1.000000 | 9.000000 | 17.000000 | 15.000000 | 16.000000 | 17.000000 | 17.000000 | 17.000000 | 9.000000 |
| **max** | 1.082873e+06 | 3.000000 | 9.000000 | 21.000000 | 18.000000 | 18.000000 | 18.000000 | 18.000000 | 25.000000 | 595.000000 |

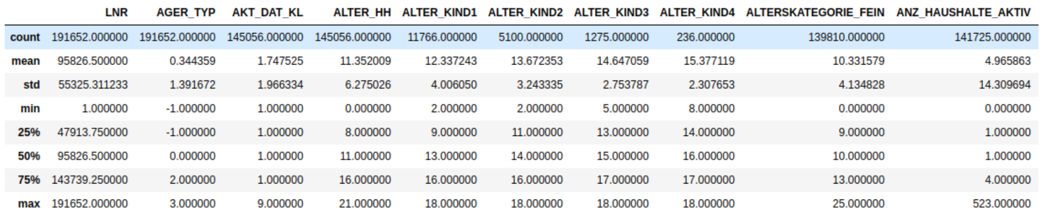
Figure 2: Descriptive statistics of general population

*Udacity_CUSTOMERS_052018.csv*: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns). Compared to *Udacity_AZDIAS_052018.csv*, 3 more features are added in this dataset: CustomerGroup, OnlinePurchase, and ProductGroup. A sample of data is shown in figure 3. Descriptive statistics of several attributes are shown in figure 4.



| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | PRODUCT_GROUP | CUSTOMER_GROUP | ONLINE_PURCHASE |
|---|---|---|---|---|---|---|---|
| **0** | 9626 | 2 | 1.0 | 10.0 | COSMETIC_AND_FOOD | MULTI_BUYER | 0 |
| **1** | 9628 | -1 | 9.0 | 11.0 | FOOD | SINGLE_BUYER | 0 |
| **2** | 143872 | -1 | 1.0 | 6.0 | COSMETIC_AND_FOOD | MULTI_BUYER | 0 |
| **3** | 143873 | 1 | 1.0 | 8.0 | COSMETIC | MULTI_BUYER | 0 |
| **4** | 143874 | -1 | 1.0 | 20.0 | FOOD | MULTI_BUYER | 0 |

Figure 3: Sample data for customers

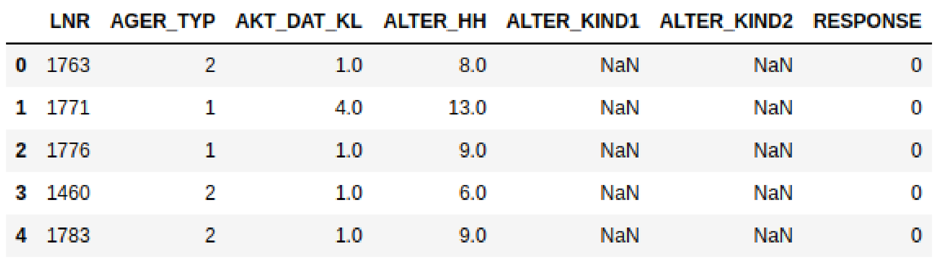


| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTERSKATEGORIE_FEIN | ANZ_HAUSHALTE_AKTIV |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 191652.000000 | 191652.000000 | 145056.000000 | 145056.000000 | 11766.000000 | 5100.000000 | 1275.000000 | 236.000000 | 139810.000000 | 141725.000000 |
| **mean** | 95826.500000 | 0.344359 | 1.747525 | 11.352009 | 12.337243 | 13.672353 | 14.647059 | 15.377119 | 10.331579 | 4.965863 |
| **std** | 55325.311233 | 1.391672 | 1.966334 | 6.275026 | 4.006050 | 3.243335 | 2.753787 | 2.307653 | 4.134828 | 14.309694 |
| **min** | 1.000000 | -1.000000 | 1.000000 | 0.000000 | 2.000000 | 2.000000 | 5.000000 | 8.000000 | 0.000000 | 0.000000 |
| **25%** | 47913.750000 | -1.000000 | 1.000000 | 8.000000 | 9.000000 | 11.000000 | 13.000000 | 14.000000 | 9.000000 | 1.000000 |
| **50%** | 95826.500000 | 0.000000 | 1.000000 | 11.000000 | 13.000000 | 14.000000 | 15.000000 | 16.000000 | 10.000000 | 1.000000 |
| **75%** | 143739.250000 | 2.000000 | 1.000000 | 16.000000 | 16.000000 | 16.000000 | 17.000000 | 17.000000 | 13.000000 | 4.000000 |
| **max** | 191652.000000 | 3.000000 | 9.000000 | 21.000000 | 18.000000 | 18.000000 | 18.000000 | 18.000000 | 25.000000 | 523.000000 |

Figure 4: Descriptive statistics of customers

*Udacity_MAILOUT_052018_TRAIN.csv*: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns). Compared to *Udacity_AZDIAS_052018.csv*, the information about the reaction to the mailout campaign is added (see figure 5).

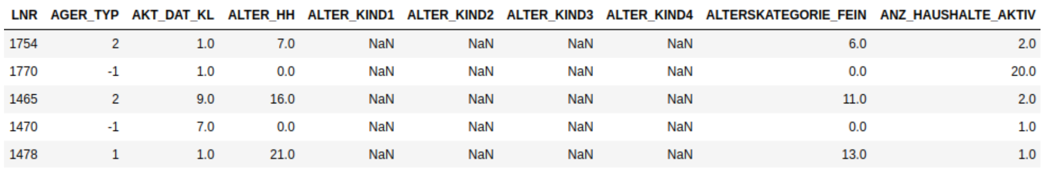
*Udacity_MAILOUT_052018_TEST.csv*: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366



| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | RESPONSE |
|---|---|---|---|---|---|---|---|
| **0** | 1763 | 2 | 1.0 | 8.0 | NaN | NaN | 0 |
| **1** | 1771 | 1 | 4.0 | 13.0 | NaN | NaN | 0 |
| **2** | 1776 | 1 | 1.0 | 9.0 | NaN | NaN | 0 |
| **3** | 1460 | 2 | 1.0 | 6.0 | NaN | NaN | 0 |
| **4** | 1783 | 2 | 1.0 | 9.0 | NaN | NaN | 0 |

Figure 5: Sample of training data

(columns). A sample of test data is shown in figure 6. Same features as in *Udacity_AZDIAS_052018.csv*.

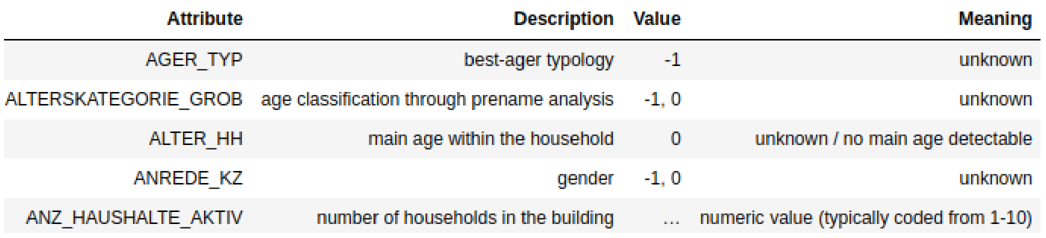


| LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTERSKATEGORIE_FEIN | ANZ_HAUSHALTE_AKTIV |
|---|---|---|---|---|---|---|---|---|---|
| 1754 | 2 | 1.0 | 7.0 | NaN | NaN | NaN | NaN | 6.0 | 2.0 |
| 1770 | -1 | 1.0 | 0.0 | NaN | NaN | NaN | NaN | 0.0 | 20.0 |
| 1465 | 2 | 9.0 | 16.0 | NaN | NaN | NaN | NaN | 11.0 | 2.0 |
| 1470 | -1 | 7.0 | 0.0 | NaN | NaN | NaN | NaN | 0.0 | 1.0 |
| 1478 | 1 | 1.0 | 21.0 | NaN | NaN | NaN | NaN | 13.0 | 1.0 |

Figure 6: Sample of test data

The features are described by two excel sheets:

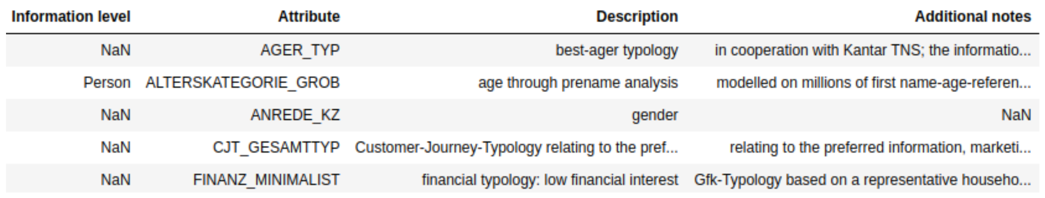
*DIAS Attributes.xlsx*: Attributes name, the description, the values and the meaning of the features (see figure 7).



| Attribute | Description | Value | Meaning |
|---|---|---|---|
| AGER_TYP | best-ager typology | -1 | unknown |
| ALTERSKATEGORIE_GROB | age classification through prename analysis | -1, 0 | unknown |
| ALTER_HH | main age within the household | 0 | unknown / no main age detectable |
| ANREDE_KZ | gender | -1, 0 | unknown |
| ANZ_HAUSHALTE_AKTIV | number of households in the building | … | numeric value (typically coded from 1-10) |

Figure 7: Sample of DIAS Attributes

*DIAS Information Levels.xlsx*: More detailed information on the features (see figure 8).



| Information level | Attribute | Description | Additional notes |
|---|---|---|---|
| NaN | AGER_TYP | best-ager typology | in cooperation with Kantar TNS; the informatio... |
| Person | ALTERSKATEGORIE_GROB | age through prename analysis | modelled on millions of first name-age-referen... |
| NaN | ANREDE_KZ | gender | NaN |
| NaN | CJT_GESAMTTYP | Customer-Journey-Typology relating to the pref... | relating to the preferred information, marketi... |
| NaN | FINANZ_MINIMALIST | financial typology: low financial interest | Gfk-Typology based on a representative househo... |

Figure 8: Sample of Information Levels

# 3 Data Preprocessing

There are a lot of missing data and lack of feature explanation in the given Excel spreadsheets. Therefore, feature extraction and data cleaning is needed. This is the most crucial step in data mining, because it determines the quality of the trained model. This process starts after scraping the data, and it consists of data cleaning, feature extraction, feature selection and feature transformation.

The data description provided by Arvato-Bertelsmann is used to build a feature summary data set. It describes the information level of features, their respective types, and which values are missing.

## 3.1 Missing Values

First step in the pre-processing phase is to deal with missing values. There are a number of ways to handle missing values including deleting rows with null values altogether, replacing null values with the mean/median/mode, replacing null values with a new category, predicting the values, or using machine learning models that can deal with null values.

Using the attributes feature summary, the attributes with unknown meaning are investigated. All the missing values are converted to NaN before checking the distribution of missing values per each column. Most attributes contain less than 40% of missing values, therefore all the features having more than 40 percent of missing values are dropped. Records that contain many missing values are harmful to machine learning algorithms. Therefore, all the records which contain more than 50% of missing values are dropped.

In the last step, analysing the meaning of the attributes, three additional columns are to be dropped because they add no value to the model:

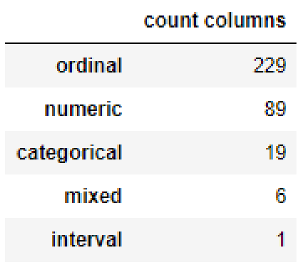
**LNR:** Unique identifier per each person.

**EINGEFUEGT_AM:** A timestamp variable.

**D19_LETZTER_KAUF_BRANCHE:** Description of other variables.

## 3.2 Feature Selection

Figure 9 shows the number of features in each data type.



| | count columns |
|---|---|
| ordinal | 229 |
| numeric | 89 |
| categorical | 19 |
| mixed | 6 |
| interval | 1 |

Figure 9: Number of attributes per each data type

All the information must be encoded numerically. Several approaches are used to re-encode features:

1. Numerical, ordinal and interval data are already numerical features so they do not need to be encoded.

2. In case of categorical attributes, one of the following approaches is used:

a) For binary numerical attributes, there is no need to encode them.

b) For binary non-numerical attributes, the values will be re-encoded as numbers.

c) For multi-level categorical variables, the values are one-hot encoded.

3. The rest of features are encoded as following:

PRAEGENDE_JUGENDJAHRE is transformed into two variables: an interval-type variable for decades, and a binary variable for movement.

LP_LEBENSPHASE_FEIN contains information about life stage and fine scale. Therefore, two new ordinals features are created: one for life_stage and other for fine_scale.

LP_LEBENSPHASE_GROB is dropped from the data as it contains the same information as LP_LEBENSPHASE_FEIN.

WOHNLAGE and PLZ8_BAUMAX are one-hot encoded.

All the steps described above are applied to all the data sets as a pre-processing pipeline. It is crucial that both the training and the testing sets to undergo under the same pre-processing steps.

## 3.3 Feature Transformation

High-dimensional data hurts a machine learning algorithm. Some features might be error-prone and will require unnecessary computational power to fit them in the training pipeline. Different set of rules can be used to reduce the number of dimensions in such a way that a high variance of information can be described in less dimensions.

One of these techniques is PCA[1]. PCA performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized. An essential parameter for PCA is the number of components to keep in the data.

Before applying PCA, all the missing data were replaced by their respective feature median values, since most of the variables have an ordinal nature. Next, feature scaling is performed. It is useful to standardize or normalize the data so that different scales of different variables don't negatively impact the performance of the model.

To conclude, a set of PCA components is computed and its respective ratio of variance is checked per each component. The cumulative variance explained is shown in the figure below:

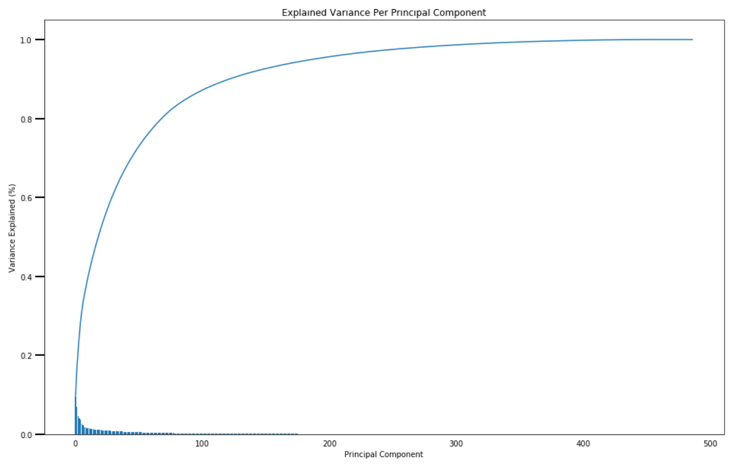


Figure 10: Explained Variance per Principle Component

Looking at the plot above, we may notice that 200 components explain

95% of the variance. In conclusion, there will be 200 dimensions kept in the data.

## 4 Model Training

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends in the data [3]. Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system:

- **Supervised Learning**: A set of feature inputs and their desired outputs are available in advance. The goal is to learn a general rule that maps inputs to outputs.
- **Unsupervised Learning**: No labels are given to the learning algorithm, leaving it on its own to find structure in its input.

### 4.1 Clustering

#### 4.1.1 K-Means

K-Means is a popular partition method which aims to cluster a set of n observations into k clusters, in which each observation belongs to the cluster with the nearest mean [4]. The centroid (e.g. mean, mode, median) is set to be the cluster representative. The main objective of the K-Means is to minimize the within-cluster variation of each clustered group. The K-Means algorithm is implemented in 2 main steps:

- **Initialization**: Choose k arbitrary representatives.
- **Repeat until representatives do not change**:
  1. Assign each object to the cluster with the nearest representative.
  2. Compute the centroids of the clusters of the current partitioning.

List of parameters defining the algorithm:

- **k**: Number of desired clusters.
- **maxIterations:** the maximum number of iterations to run.
- **initializationMode:** specifies either random initialization or initialization via k-means++ [5].

### 4.1.2 DBSCAN

According to DBSCAN, clusters are dense regions in the data space, separated by regions of lower object density [6]. For any point in the cluster, the local point density around that point has to exceed some threshold. The set of points from one cluster is spatially connected. In DBSCAN, points are classified as core points, density-reachable points and outliers. A point q is a core point if at least a set of predefined points are within the distance *epsilon*. These points are said to be directly reachable. All the points not reachable from any other point are outliers. Local point density at point q is defined by two parameters:

- **epsilon**: radius for the neighborhood of the studied point.
- **Minimum Number of Points**: minimum number of points in the given neighborhood.

### 4.1.3 Optimization Techniques

Determining the most optimal number of clusters is a frequent problem in data clustering, and is a distinct issue from the process of actually solving the clustering problem. For a certain class of clustering algorithms such as KMeans, this parameter is referred to K. On density-based algorithms, a set of different parameters must be defined in advance: *epsilon, min_samples.* Several techniques how to properly determine these parameters are explained below.

- **Silhouette Coefficients**: This technique displays a measurement of how close each point in one cluster (a(i)- cohesion) is to points in the neighboring cluster (b(i) -separation), and thus provides a way to assess parameters like number of clusters visually. The silhouette ranges from

-1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters: $s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$.

The silhouette of a cluster C is defined as: $s_c = \frac{1}{n_c}\sum_{o\epsilon C} s(o)$. The silhouette coefficient of a cluster C ranges between 0 and 1. A strong model is considered when: $0.7 < s_c < 1$.

- **Elbow Method**: Another method to define K is the Elbow method. The idea of the elbow method is to run KMeans clustering on the dataset for a range of values of K (say K from 2 to 26 as in figure 11), and for each value of K calculate the sum of squared errors (SSE) or within cluster sum of squared errors (WCSS) [7].

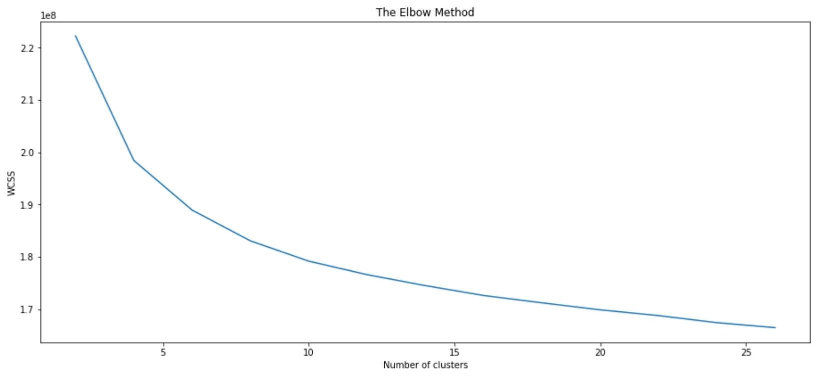


Figure 11: Elbow Method

WCSS is the summation of the each clusters distance between that specific clusters each points against the cluster centroid:

$WCSS = \sum_{P_i inCluster1} distance\,(P_i, C_1)^2 + \sum_{P_i inCluster2} distance\,(P_i, C_2)^2 +$ ...

Then, plot a line chart of the WCSS for each value of K. If the line chart looks like an arm, then the "elbow" on the arm is the value of K that is the best. The idea is that we want a small WCSS, but that the WCSS tends to decrease toward 0 as we increase K (the WCSS is 0 when K is equal to number of data points in the dataset). So the goal is to choose a small value of K that still has a low WCSS, and the elbow usually represents where we start to have diminishing returns by increasing k. From the example attached above, the most optimal number of clusters results to be K=6.

### 4.1.4 Advantages and Disadvantages of each Algorithm

Table 1 briefly summarizes the advantages and disadvantages of each algorithm.

Table 1: Pros and Cons of each Algorithm

| | **Advantages** | **Disadvantages** |
|---|---|---|
| **K-Means** | 1. Relatively efficient: O(tkn), where n = number of objects, k = number of clusters, and t = number of iterations.<br>2. Typically: k, t smaller than n.<br>3. Easy implementation. | 1. Applicable only when mean is defined.<br>2. Specify the number of cluster in advance.<br>3. Sensitive to noisy data and outliers. |
| **DBSCAN** | 1. Clusters can have arbitrary shape and size, i.e. clusters are not restricted to have convex shapes.<br>2. Number of clusters is determined automatically.<br>3. Can separate clusters from surrounding noise.<br>4. Can be supported by spatial index structures. | 1. Input parameters may be difficult to determine.<br>2. In some situations very sensitive to input parameter setting. |

### 4.1.5 Results

Evaluating clustering results is a difficult process because of missing labels. However, given the pros and cons above, some conclusions can be drawn. KMeans is selected to perform clustering because it is pretty efficient and easy to implement. DBSCAN on the other hand is widely used for anomaly detection problems, which is not our specific case. Additionally, DBSCAN has difficulties determining the input parameters. For better visualization purposes, the elbow method is additionally selected to determine the most optimal number of clusters.

From the elbow method (see figure 11), we can conclude that the most optimal number of clusters is 6. The results of running KMeans in 6 clusters are visualized in figure 12. To sum up, 99.9% of the customers' data can be represented by cluster 2, which contains 31% of the general population. Therefore, people belonging to cluster 2 have a higher probability being future customers of the company.

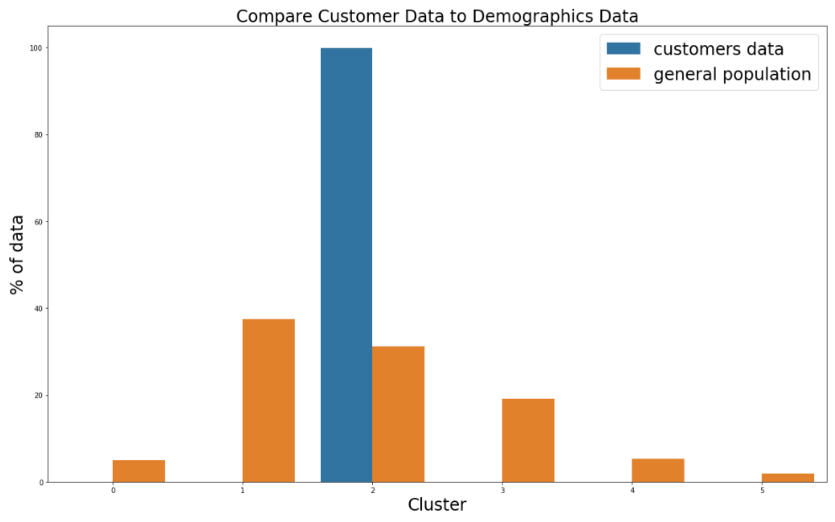


Figure 12: Distribution of customers' data in each cluster

## 4.2 Classification

Nest step in this project is to build a predictive model, which returns a response score for each customer. The demographics data of individuals who were targets of a marketing campaign will be used for training the model. This data contains a dependent variable (Response), which indicates if the customer responses positively or negatively to the campaign. The same cleaning and transformation steps as in the previous section have to be performed before training the model.

### 4.2.1 Data Exploration

Out of 42 962 individuals in the training data, only 532 people response positively to the campaign (see figure 13). The training dataset is highly imbalanced.

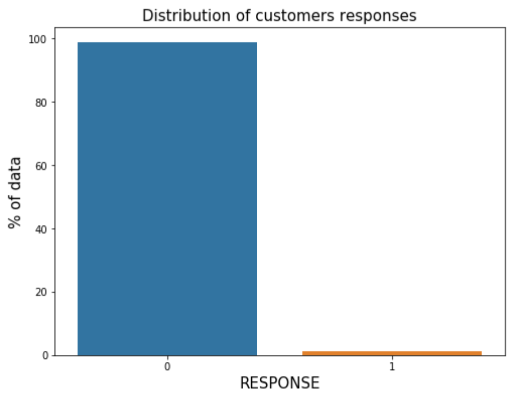


Figure 13: Distribution of Response

In general, to build a supervised model, it is required to split data into training and testing sets. In our specific case, there is a test dataset provided. The performance of the model can be evaluated by submitting the test predictions to the Kaggle competition.

### 4.2.2 Classifiers

Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone[8]. The ensemble models can be used for both classification and regression problems. They tend to improve the stability and the accuracy of the models, have higher consistency and reduce overfitting. There are two main approaches in ensemble learning: bagging and boosting. In this project, the focus is in boosting since it tends to teach the ensemble model.

Boosting is an ensemble method to improve a model by reducing its bias and variance, ultimately converting weak learners to strong learners. The general idea is to train a weak learner and sequentially iterate and improve the model by learning from the previous learner.

While training the model, cross validation will be used because there is only 1.2% of the customers responding positively to the campaign. This number is pretty low to further split the data. By default, GridSearchCV[2] uses 5-fold cross validation to obtain the learning curves.

Several ensemble models are trained with default parameters:

**1. Random Forest Classifier[3]:** Random Forest is an ensemble method that uses many weak decision trees to make a strong learner. Random forests are more accurate, more robust, and less prone to overfitting compared to decision trees.

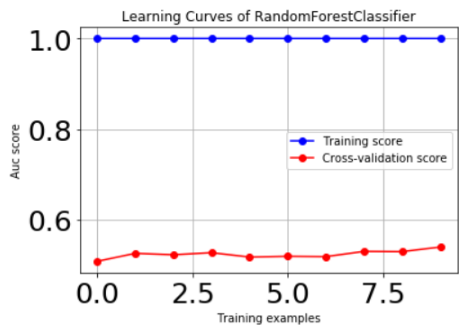


Figure 14: Learning Curves of Random Forest

**2. Adaboost Classifier[4]:** An estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

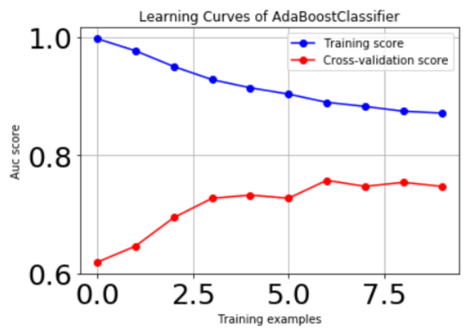


Figure 15: Learning Curves of Adaboost

[2]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV
[3]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier
[4]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier

**3. Gradient Boosting Classifier**[5]**:** Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

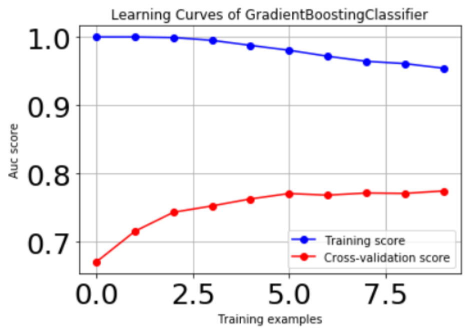


Figure 16: Learning Curves of Gradient Boosting

**4. XGBoost Classifier**[6]**:** XGBoost stands for Extreme Gradient Boosting; it is a specific implementation of the Gradient Boosting method which uses more accurate approximations to find the best tree model. For example, it uses advanced regularization (L1 & L2), which improves model generalization. Furthermore, XGBoost uses the 2nd derivative as a proxy for minimizing the error.

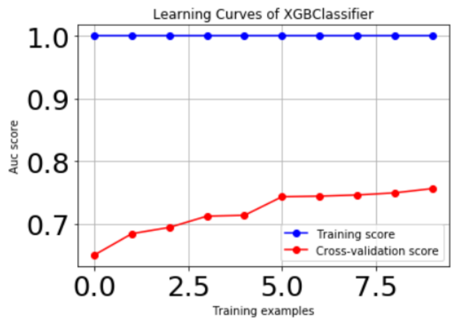


Figure 17: Learning Curves of XGBoost

[5]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier
[6]https://xgboost.readthedocs.io/en/latest/python/python_api

#### 4.2.3 Benchmark

To determine how different models affect the predictions quality and to check what's the impact of hyper-parameter tuning, a benchmark model is selected. From the models explained above, Random Forest is the simplest one. It fits a classifier to a random target vector of 0s and 1s and counts how many times the observations fall into the same node. Random Forest Classifier is not the most suitable estimator for imbalanced datasets, therefore it will be selected as a benchmark model. As observed in figure 14, Random Forest overfits the training data as it has almost no error on the training set and achieves a score of 0.5 on the validation set. Let's check how these results can be improved by selecting better estimators and by tuning their hyper-parameters.

#### 4.2.4 Evaluation

Since the data is highly imbalanced, the ROC-AUC score[7] will be used to evaluate the model's performance. A learning curve shows the training and validation score of an estimator for varying numbers of training samples. Cross validation starts training each model with a small subset of the training data and increases the subset size gradually. In other words, the learning curve is the comparison between the evolution of the training error and the cross-validation error at each training phase.

Analysing figure 14, the learning curves show that our benchmark model (Random Forest) is over-fitting the data. The error on the validation set is pretty high, which makes this a bias model.

Ada Boost performs better than the benchmark as it is visualized in figure 15. The training score decreases as we add more training subsets, but it increases for the validation set. The curves are almost converged and the model will not improve by adding more data in the training set.

The training score of Gradient Boosting decreases to 0.935, while the validation score increases up to 0.78 (see figure 16). The curves are yet not converged and adding more data points can improve the score.

XGBoost performs very well on the validation set, but seems to overfit the data on the training set (see figure 17). Therefore, we drop XGBoost in favor of Gradient Boost, even though in theory XGBoost is a better model since it adds some approximations to find the best tree model.

[7]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score

Compared to the benchmark model, all the models perform better on the validation set. Gradient Boosting results to be the most efficient estimator. The validation score is increased from 0.5 in the benchmark to 0.78 in Gradient Boosting. On the other hand, the training score of Gradient Boosting is decreased, but this is a good sign, showing that the model is not over-fitting the data.

### 4.2.5 Parameter Tuning

To tune the hyper-parameters, Grid Search is used. Grid Search tests all possible combination of specific hyper-parameters using cross validation. Then, it selects the model with the maximum roc_auc score on the validation set. After tuning the Gradient Boosting estimator, the following hyper-parameters are calculated: learning_rate=0.05, n_estimators=100, decision tree max_depth=3 and min_sample_split=2. The roc_auc score of the tuned model increases to 0.98 on the training set, compared to 0.93 in the default Gradient Boosting model. The only parameter to be changed from the default values of Gradient Boosting is the learning_rate (default value = 0.1 vs tuned value = 0.05).

From figure 18, the most important feature is D19_SOZIALES, significantly having a higher weight than the second most important feature which is KBA13_ANZAHL_PKW.

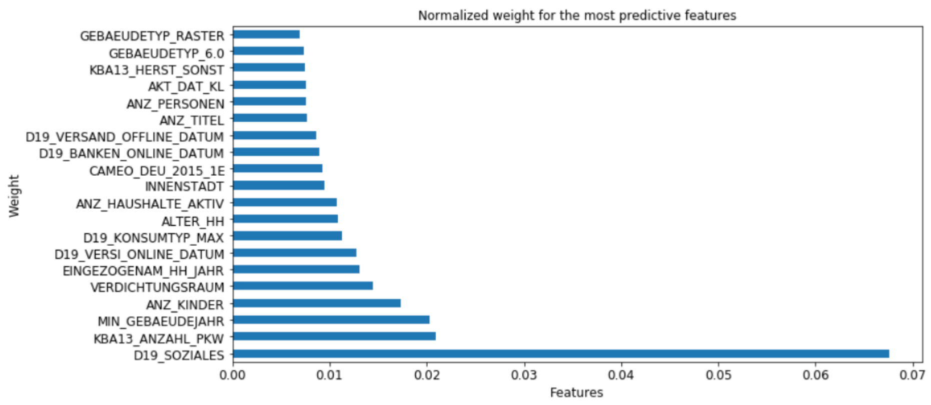


Figure 18: Features Importance

## 5 Kaggle Competition

The tuned Gradient Boosting model is finally applied to the provided test data. The results are submitted to Kaggle competition, where the final roc_auc score is 0.79068. Cross Validation and Grid Search helped improving the model quality. However, with better domain knowledge and feature engineering, higher results can be achieved.

## 6 Conclusion

In this project, supervised and unsupervised learning techniques were used to analyze demographics data for customers of a mail-order company (Arvato Financial Solutions) in Germany against demographics information of the existing clients. Different stages of the data science process were followed to solve the problem of customer acquisition.

In the first part, data was cleaned, transformed, scaled and encoded properly. This is the most important stage because it is necessary to convert the data to a proper format for the machine learning algorithms. In the second part, customer segmentation is performed using clustering algorithms (i.e. KMeans). The main goal here was to map current company customers with the most similar population cluster set. In the final step, the labels of a mail order campaign were used to build a supervised learning model that predicts whether the people will respond to the campaign. The most efficient model was Gradient Boost with a roc_auc score of 0.79068.

The model can be further improved by more hyper-parameter tuning and better feature engineering. In addition, more domain knowledge can help the pre-processing part. There might be crucial attributes which can be better engineered rather than dropped. Using more advanced algorithms (i.e. neural networks) can also improve the model performance. In such a case, more computational power (i.e. GPU) is required for fast processing.

## References

[1] Ian T. Jolliffe and Jorge Cadima. *Principal component analysis: a review and recent developments.*

[2] Celebi, M Emre and Aydin, Kemal *Unsupervised Learning Algorithms.*

[3] Wagstaff, Kiri. *Machine learning that matters.*

[4] Wagstaff, Kiri and Cardie, Claire and Rogers, Seth. *Constrained k-means clustering with background knowledge.*

[5] Arthur, David and Vassilvitskii, Sergei. *k-means++: The advantages of careful seeding.*

[6] Kantardzic, Mehmed. *Data mining: concepts, models, methods, and algorithms.*

[7] Bholowalia, Purnima and Kumar, Arvind. *EBK-means: A clustering technique based on elbow method and k-means in WSN.*

[8] Opitz and Maclin *Popular ensemble methods: An empirical study.*