Capstone Project Proposal

Machine Learning Engineer Nanodegree

Customer Segmentation – Arvato Financial Solutions

Vinicius Gomes Pereira

viniciusgomespe@gmail.com

18/04/2020

1. The Market Segmentation Problem

Market segmentation consists of identifying a group of individuals who have characteristics in common, in a heterogeneous market. This should be seen as a powerful tool for auxiliary departments of marketing, design and strategic areas.

Several characteristics of consumers and potential consumers can be taken into account for this segmentation task, for example:

1. Demographic, geographic, social and economic criteria.
2. Personality and lifestyle criteria.
3. Product behavior criteria

The question is what is the advantage of segmentation? The market is heterogeneous and therefore it is advisable to segment it. Thus, attitudes towards these segments can be different. Segmentation allows to rationalize the means to reach a given product segment, adjusting it to the prices and costs of distribution and communication, with a view to achieving balance. It also allows a specialization of the company playing with the strategic variables - price, product, distribution and communication - avoiding waste.

2. Project Overview

Consider a company's marketing campaign (Arvato Financial Services), in which we need to select those individuals who can become the company's future customers. For this task, we have the following databases: demographic information from Germany (country where the company is located) and information from individuals who are already customers of this company.

First, the demographic information of the German population was analyzed in order to understand and explore the main characteristics of this population.

Then, we create a predictive model that can determine with reasonable accuracy whether a person can become a possible consumer of the company, when subjected to a certain marketing campaign.

Finally, we classify each possible consumer, from an unexplored test database, and submit the result on the kaggle platform.

3. Problem Statement

We can define this work with the following question: “Which people are more likely to become future consumers when they are submitted to a marketing campaign?”. In this way, the work will be divided into 3 parts:

1. Exploratory Data Analysis (EDA) from the general population and consumers
2. Development and Training of Algorithms to recognize whether a person will become a consumer or not of the company, when submitted to a marketing campaign
3. Submission of this algorithm on the Kaggle platform in order to assess the efficiency of the algorithm

4. Project Design

First, we will do an exploratory pre-analysis of the data. In this part, we will not process the data, and the objective is to know the database beforehand. Then, we will follow the next framework, in the exploratory data analysis part, as shown in image 4.1:

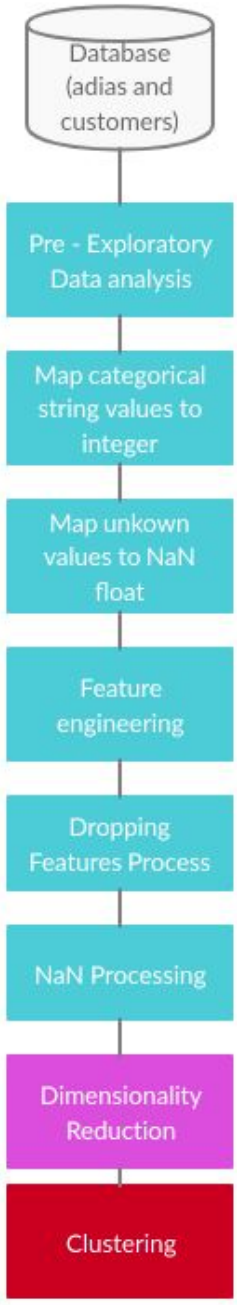


Figure 4.1 - Pre-EDA, EDA, Dimensionality Reduction and Clustering

Next, we will train several models in the training database. We will explore the following listed models:

1) XgBoost [4]
2) XgBoost with downsample of majority class
3) Baseline XgBoost (with default params of training)
4) Logistic Regression [5]
5) Logistic Regression with downsample of majority class
6) Baseline Logistic Regression (with default params of training)
7) Random Forest [6]
8) Random Forest with downsample of majority class
9) Baseline Random Forest (with default params of training)

For each model (with the exception of the baseline models), we will train considering 2 scenarios (whether or not using majority class downsample). In addition, we will use Randomized Grid Search [3] for hyperparameter tuning with stratified cross validation with 3 folds, then plot the learning curve. For each model, we will predict the test set and submit it on the Kaggle platform [1]. Then, we will analyze the performance of each model, and the most important features of the best model.

5. Evaluation Metric

Accuracy, recall and precision are common metrics to use for classification task. These three evaluation metrics are defined below, where TP, TN, FP, FN mean true positives, true negatives, false positive and false negatives in the classification process:

accuracy = (TP+ TN)/(TP + FP + TN+FN)
recall = (TP)/(TP+FN)
precision = (TP)/(TP+FP)

However, we will evaluate each of the models, using ROC-AUC (Area Under the Curve Receiver Operating Characteristics) [2], due to the great imbalance between the classes, and in order to also minimize the error in which our prediction in the set of test is the same as the evaluation of the Kaggle platform, where we will submit the answers. The ROC curve is a graphic used to plot the true positive rate TPR (true positive rate) against FPR (false positive rate).

For dimensionality reduction (PCA) [7] , we will compute the number of components that retains 95% of the variance of the original data.

For the k-Means algorithm [8] , we will going to compute the sum of squared distances of samples to their closest cluster center, and we will study the increase of the number of clusters and the decreasing of these average to decide the number of clusters of our analysis.

6. Pre - Exploratory Data Analysis

6.1 Datasets and Inputs

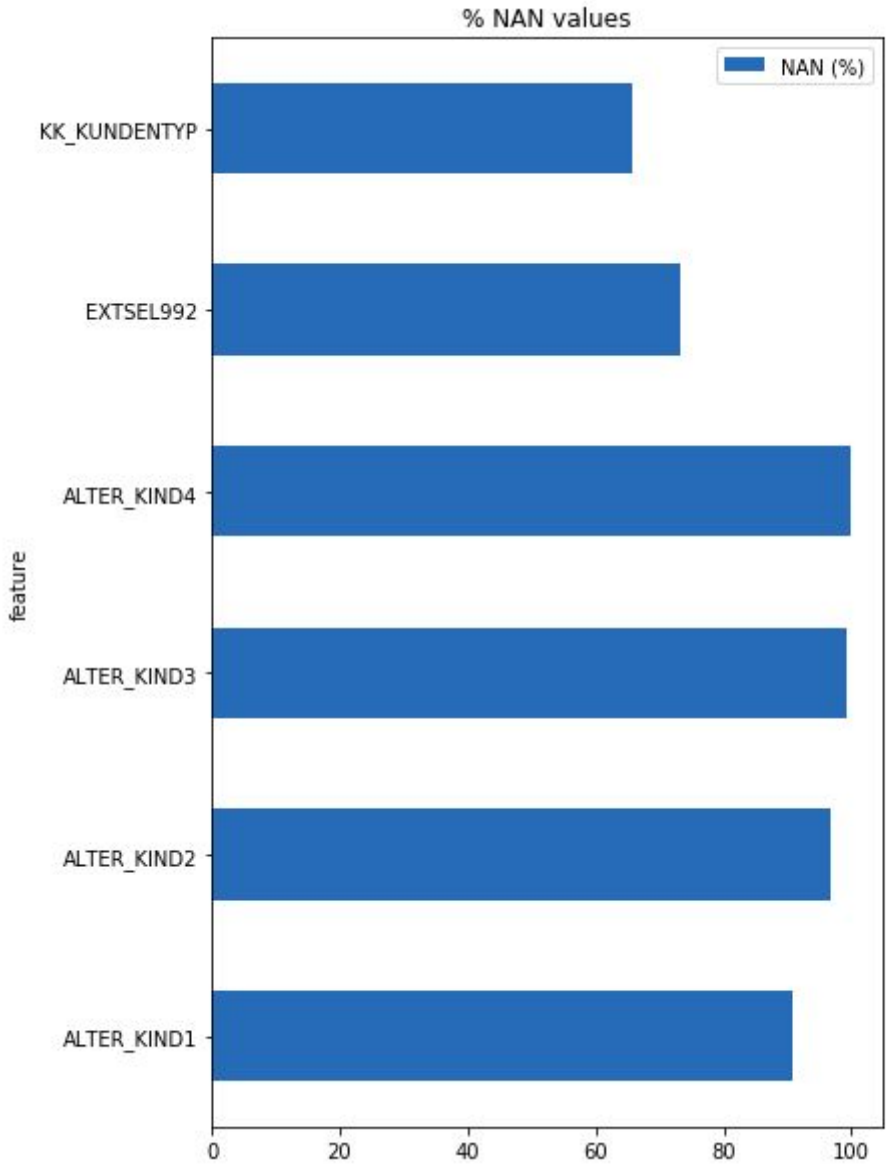
There are four data files associated with this project, that were provided by Udacity :

1. Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns);
2. Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns);
3. Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns);
4. Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns);
5. DIAS Information Levels — Attributes 2017.xlsx: Contains information about the values of some attributes, as well as their meaning

The first and the second file will be used in the customer segmentation step. The third one will be used to train the predictive model that will be used to identify possible consumers. The forth file will be used to measure how well our model performs in the classification task.

6.2 Initial Data Exploring

The data set Udacity_AZDIAS_052018 has 891211 rows and 366 features. Some features has high number of empty values. In the graph below, it is possible to see some features with more than 50% of the data with empty values.



Most of the features of Udacity_AZDIAS_052018 is categorical, only ANZ_HAUSHALTE_AKTIV, ANZ_HH_TITEL, ANZ_PERSONEN, ANZ_TITEL, GEBURTSJAHR, KBA13_ANZAHL_PKW, MIN_GEBAEUDEJAHR are of numeric types. The meaning of each numerical feature is described in the next table:

Numeric Features	
Feature	Description
ANZ_HAUSHALTE_AKTIV	number of households in the building
ANZ_HH_TITEL	number of academic title holder in building
ANZ_PERSONEN	number of adult persons in the household
ANZ_TITEL	number of professional title holder in household
GEBURTSJAHR	year of birth
KBA13_ANZAHL_PKW	number of cars in the PLZ8
MIN_GEBAEUDEJAHR	year the building was first mentioned in our database

The next table has some information of theses features, like mean, standard deviation, min and max values and percentiles. It is possible to notice , for example that a 1.72 is the average number of adults in the households and 1985 is the oldest year of a building mentioned in database.

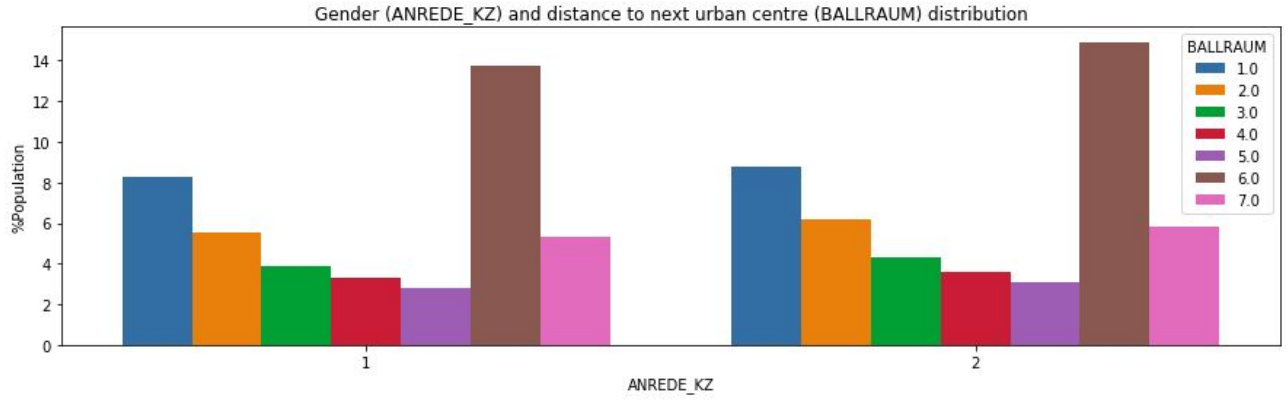
	ANZ_HAUSHALTE_AKTIV	ANZ_HH_TITEL	ANZ_PERSONEN	ANZ_TITEL	GEBURTSJAHR	KBA13_ANZAHL_PKW	MIN_GEBAEUDEJAHR
count	798073.000000	794213.000000	817722.000000	817722.000000	891221.000000	785421.000000	798073.000000
mean	8.287263	0.040647	1.727637	0.004162	1101.178533	619.701439	1993.277011
std	15.628087	0.324028	1.155849	0.068855	976.583551	340.034318	3.332739
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1985.000000
25%	1.000000	0.000000	1.000000	0.000000	0.000000	384.000000	1992.000000
50%	4.000000	0.000000	1.000000	0.000000	1943.000000	549.000000	1992.000000
75%	9.000000	0.000000	2.000000	0.000000	1970.000000	778.000000	1993.000000
max	595.000000	23.000000	45.000000	6.000000	2017.000000	2300.000000	2016.000000

Some features are categorical, for example:

Categorical Features	
Feature	Description
ALTERSKATEGORIE_GROB	age classification through prename analysis
ANREDE_KZ	gender
BALLRAUM	distance to next urban centre
D19_BANKEN_ONLINE_QUOTE_12	amount of online transactions within all transactions in the segment bank

D19_BUCH_RZ	transactional activity based on the product group BOOKS and CDS
D19_KONSUMTYP	consumption type
D19_KK_KUNDENTYP	consumption movement in the last 12 months

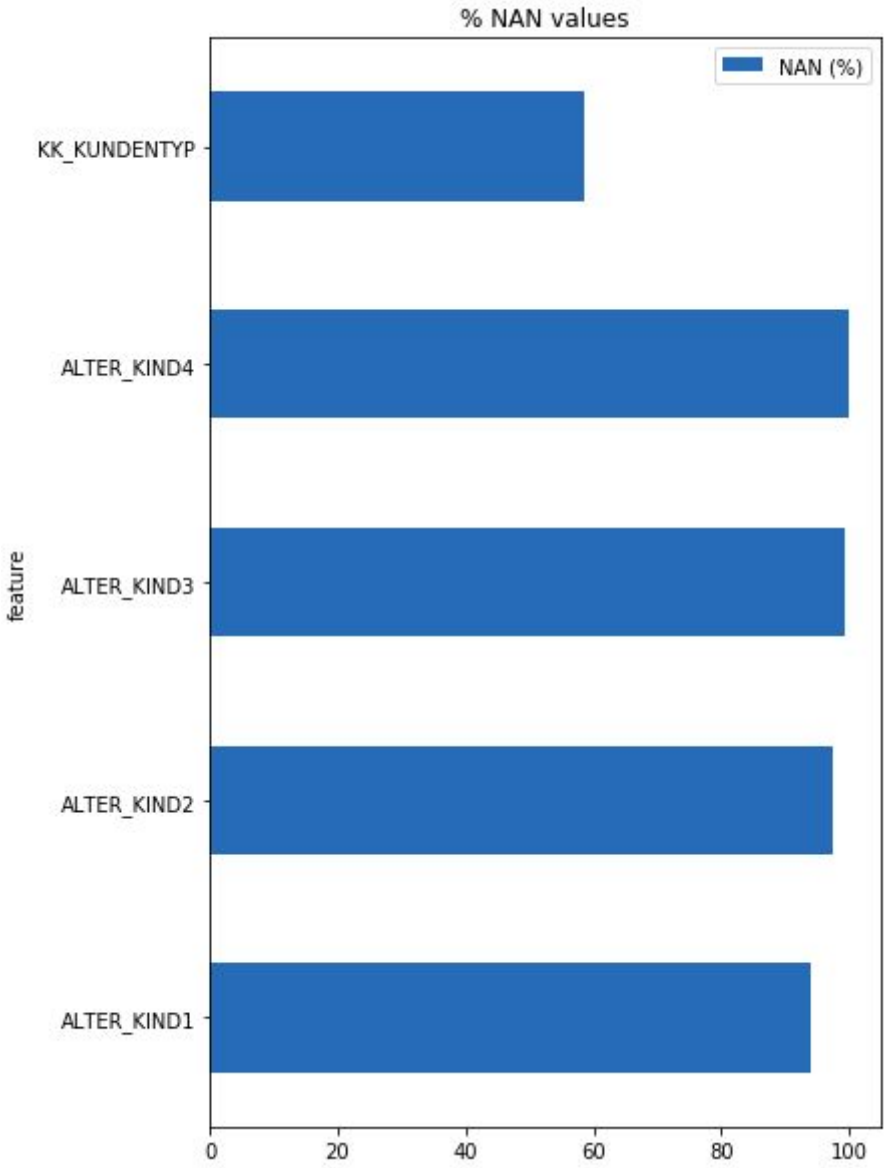
In the graph below, we can see how people are distributed from the urban centre, by gender, where de categorical value (ANREDE_AZ) 1 means ‘Male’ and 2 means ‘Female’ and BALLRAUM values are explained in the next table.



BALLRAUM	
Value	distance to next urban centre
-1	unknown
1	till 10 km
2	10 - 20 km
3	20 - 30 km
4	30 - 40 km
5	40 - 50 km
6	50-100 km
7	more than 100 km

Hence, we can conclude that most of the people lives between 50-100 km from the next urban centre and there are not significance difference of behavior analyzing this feature by gender.

The Udacity_CUSTOMERS_052018.csv Demographics data for customers of a mail-order company with 191 652 persons with 369 features. Some features has high number of empty values. In the graph below, it is possible to see some features with more than 50% of the data with empty values. The result is similar to the dataset of general population of Germany, as we previously have discussed.

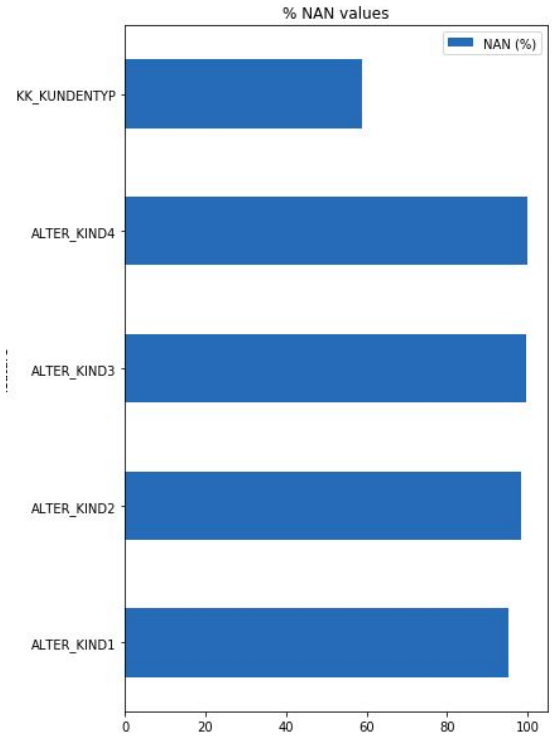


Besides, in this dataset, there are three categorical features that the first dataset does not have: ‘PRODUCT_GROUP', 'ONLINE_PURCHASE' and 'CUSTOMER_GROUP', with information about the person behavior as a consumer. The next table has the dataset of consumers grouped in theses three features:

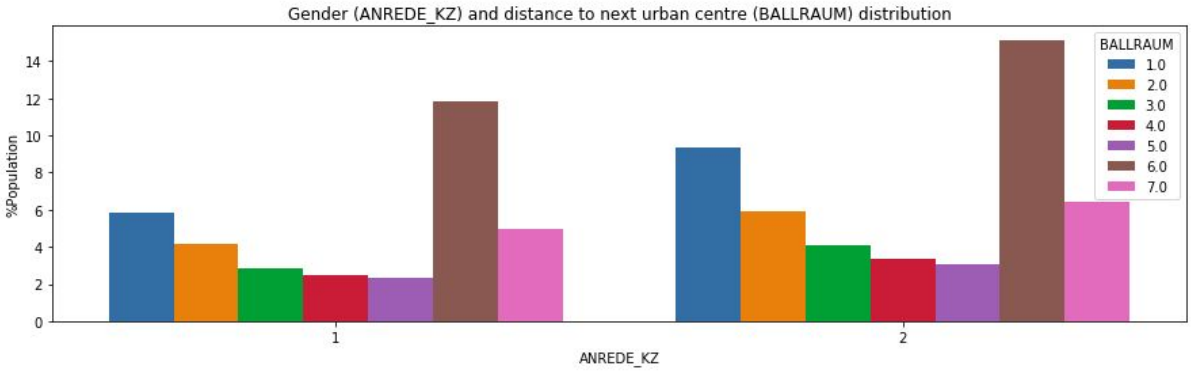
	PRODUCT_GROUP	ONLINE_PURCHASE	CUSTOMER_GROUP	COUNT	%CONSUMERS
0	COSMETIC	0	MULTI_BUYER	17105	8.925031
1	COSMETIC	0	SINGLE_BUYER	21022	10.968839
2	COSMETIC	1	MULTI_BUYER	1983	1.034688
3	COSMETIC	1	SINGLE_BUYER	3300	1.721871
4	COSMETIC_AND_FOOD	0	MULTI_BUYER	92941	48.494667
5	COSMETIC_AND_FOOD	1	MULTI_BUYER	7919	4.131968
6	FOOD	0	MULTI_BUYER	11054	5.767746
7	FOOD	0	SINGLE_BUYER	32234	16.819026
8	FOOD	1	MULTI_BUYER	1236	0.644919
9	FOOD	1	SINGLE_BUYER	2858	1.491245

In both data sets, we will perform the exploration data analysis, using PCA and K-Means.

Udacity_MAILOUT_052018_TRAIN.csv has demographics data for individuals who were targets of a marketing campaign with 42982 persons and 367 features. There is one additional column in relation to Udacity_AZDIAS_052018, with the variable response. Some features has high number of empty values. In the graph below, it is possible to see some features with more than 50% of the data with empty values.



In the next graph is plotted how people are distributed from the urban centre, by gender. It is quite similar to the first data set of population of Germany. We can conclude , for example, that most of the people lives between 50-100 km from the next urban centre and there are not significance difference of behavior analyzing this feature by gender.



The data set Udacity_MAILOUT_052018_TEST.csv has demographics data for individuals who were targets of a marketing campaign with 42 833 persons and 366 features in columns. This is the data set that we will perform classification.

7. EDA

7.1 Preprocessing

Some features contain string values and need to be mapped to integer values. like OST_WEST_KZ, CAMEO_DEU_2015 and D19_LETZTER_KAUF_BRANCHE. In addition, the EINGEFUEGT_AM feature represents a date and we map it to the corresponding year of the date. The features CAMEO_DEUG_2015 and CAMEO_INTL_2015, which represent whole values and are categorical came with unexpected values according to the dictionary of attributes: some lines came with 'XX'. That way, we are going to mapping these values to null values as well.

In this preprocessing phase, we also eliminate columns that do not have information such as "Unnamed: 0" and LNR.

7.2 Mapping Unknown Values

Using the dictionary of attributes, we can identify which values in the database are filled in but represent null values. Thus, previous analysis does not take this mapping into account. Therefore, the next step in our analysis is to map the filled data, which corresponds to null values, and update them to a really null value. The table below illustrates the attributes and their respective null values. There is even the occurrence of more than one possible null value for a given attribute, such as: ALTERSKATEGORIE_GROB

Attribute	Description	Unknown
AGER_TYP	best-ager typology	-1
ALTERSKATEGORIE_GROB	age classification through prename analysis	-1, 0
ALTER_HH	main age within the household	0
ANREDE_KZ	gender	-1, 0
BALLRAUM	distance to next urban centre	-1
BIP_FLAG	business-flag indicating companies in the building	-1
CAMEO_DEUG_2015	CAMEO classification 2015 - Uppergroup	-1
CAMEO_DEUINTL_2015	CAMEO classification 2015 - international typology	-1
CJT_GESAMTTYP	customer journey typology	0

7.3 Feature Engineering and Droping Columns Process

Many features have been created, and some have been transformed. Some features had information of mixed natures, such as WOHNLAGE, which has the information if the person lives in an urban or rural place and also if the person lives in a place of good or bad quality.

Another example of a feature that aggregates a variety of information is LP_STATUS_GROB, which has information on whether the person is single, married, etc., as well as information on whether they have good financial conditions.

Another different strategy in this feature extraction. was to decrease the number of classes. For example, the D19_VERSAND_ONLINE_QUOTE_12 feature has 11 classes, which means the percentage of online transactions you have carried out in the last 12 months (0%, 10%, 20%, etc.). We reduced that number to just 5 classes (0%, 10% -30%, 40% -60%, 70% -90%, 100%).

Thus, the following table shows how the original feature was transformed into a new feature.

Original Feature	New Feature
CAMEO_DEU_2015	CAMEO_DEU_2015 represents many categories from CAMEO classification. It is represented by a number and a letter. We take the letter and mapped to an integer number.
CAMEO_DEUG_2015	From CAMEO_DEUG_2015, it is possible to extract the social status. So, we identified the upper class, the middle class, the lower class and the working class.
CAMEO_INTL_2015	From CAMEO_INTL_2015 it is possible to extract the wealth Status of the person: wealthy, prosperous, comfortable, etc
CAMEO_INTL_2015	From CAMEO_INTL_2015 it is possible to extract the life Staging Of the person: couple, single, with children, etc
PRAEGENDE_JUGENDJAHRE	From PRAEGENDE_JUGENDJAHRE it is possible to extract the generation of the person: forties, fifties, etc
PRAEGENDE_JUGENDJAHRE	From PRAEGENDE_JUGENDJAHRE it is possible to extract the generation movement of the person: mainstream/avantgarde
LP_LEBENSPHASE_FEIN	From LP_LEBENSPHASE_FEIN it is possible to extract the life stage: younger age, etc
LP_LEBENSPHASE_FEIN	From LP_LEBENSPHASE_FEIN it is possible to extract the life fine: low, average, wealthy and top
WOHNLAGE	From WOHNLAGE it is possible to extract where the person lives
WOHNLAGE	From WOHNLAGE it is possible to extract the quality of the place where the person lives
LP_STATUS_GROB	We Divide the group into low-income earners, average earners, independants, house Owners, top earners
LP_FAMILIE_GROB	We Divide the group into single , couple, single parent, family, and multi person household
D19_VERSAND_ONLINE_QUOTE_12	We Divide the group of online transactions into only 5 groups (0, 10%-30%, 40%-60% , 70%-90% and 100%)
D19_BANKEN_ONLINE_QUOTE_12	We Divide the group of online transactions into only 5 groups (0, 10%-30%, 40%-60% , 70%-90% and 100%)
D19_GESAMT_ONLINE_QUOTE_12	We Divide the group of online transactions into only 5 groups (0, 10%-30%, 40%-60% , 70%-90% and 100%)
D19_BANKEN_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)

D19_BANKEN_OFFLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, Increasing activity and activity elder than 1 year)
D19_BANKEN_ONLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_GESAMT_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_GESAMT_OFFLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_GESAMT_ONLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_TELKO_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_TELKO_OFFLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_TELKO_ONLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_VERSAND_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_VERSAND_OFFLINE_DATUM	We Divide the group of activity into only 4 groups (no activity, high activity, increasing activity and activity elder than 1 year)
D19_VERSI_ANZ_12	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_VERSI_ANZ_24	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_BANKEN_ANZ_12	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_BANKEN_ANZ_24	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_GESAMT_ANZ_12	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_GESAMT_ANZ_24	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_TELKO_ANZ_12	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_TELKO_ANZ_24	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)
D19_VERSAND_ANZ_12	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)

D19_VERSAND_ANZ_24	We Divide the group of transactions into only 4 groups (low, high, medium and none activity)

After this process some features were created and others were transformed. In graph 7.3.1, we show a histogram of the percentage of null values in the columns, of the dataset of the general population. The vast majority of columns have less than 30% of their values null. In graph 7.3.2, we also show the same histogram, with similar behavior, but from the consumer dataset.

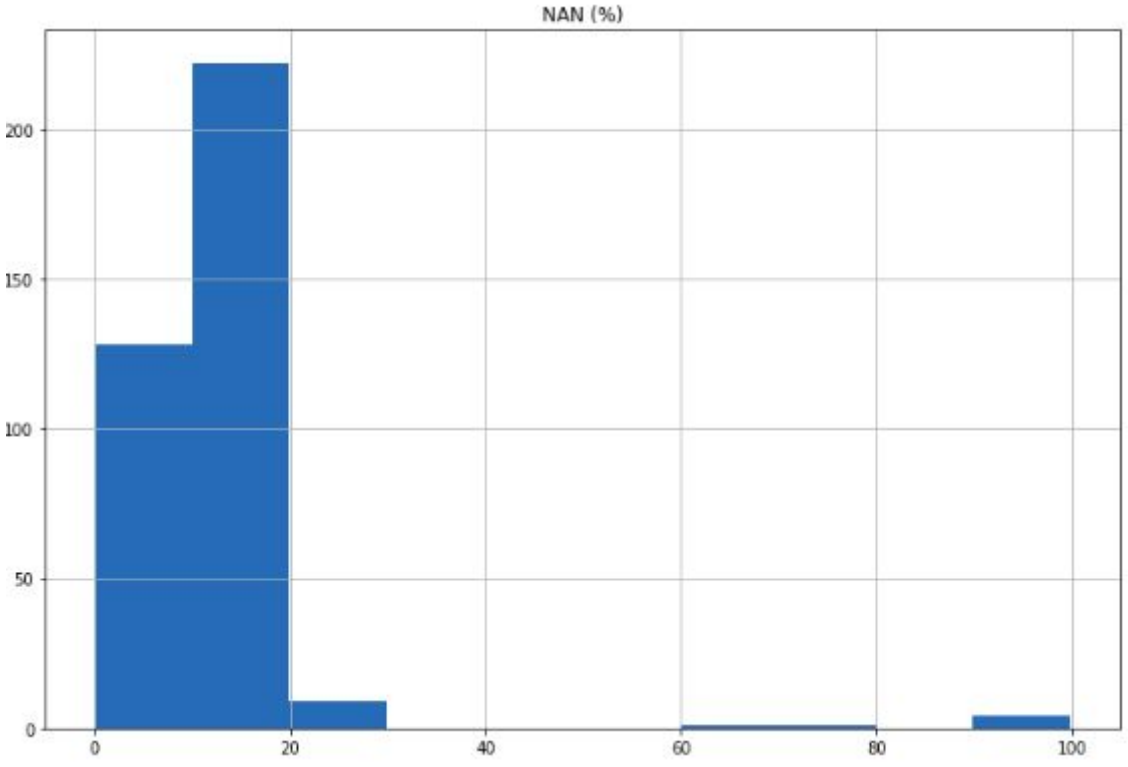


Figure 7.3.1 - Histogram of the proportion of null values of the features of the AZDIAS data set

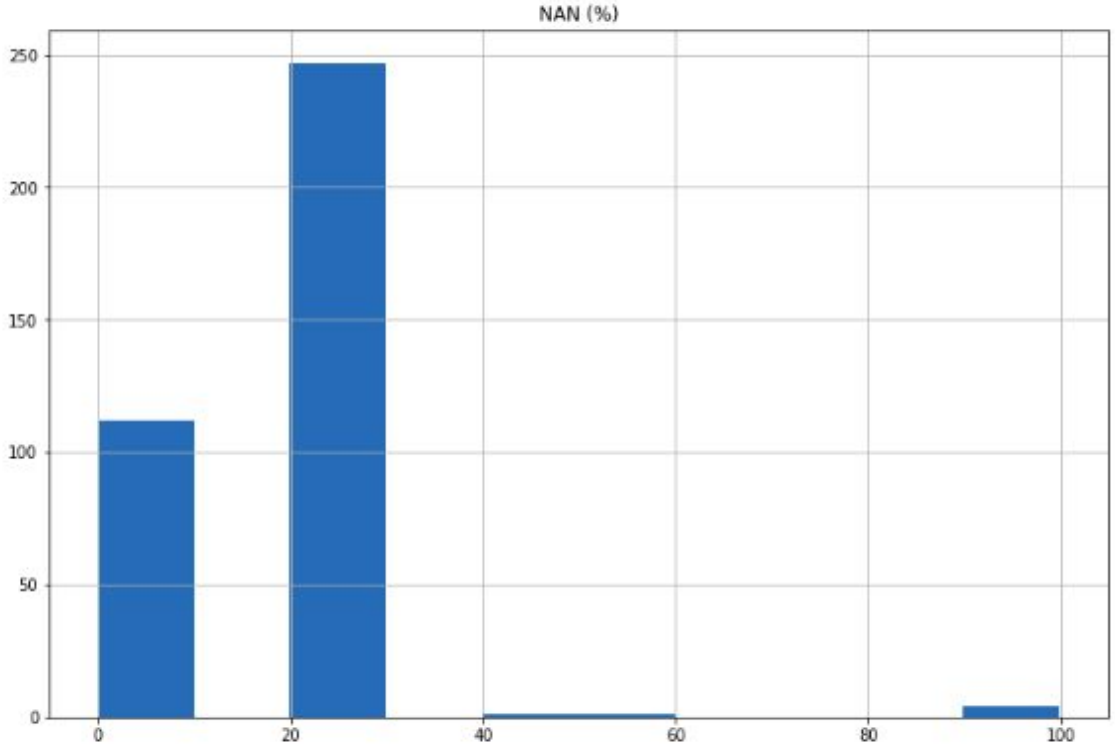


Figure 7.3.2 - Histogram of the proportion of null values of the features of the Consumers data set

In graph 7.3.3, we show which features have more than 60% of their null values, in the general population data set. In item 7.3.4, the same item is shown, however in the consumer data set.

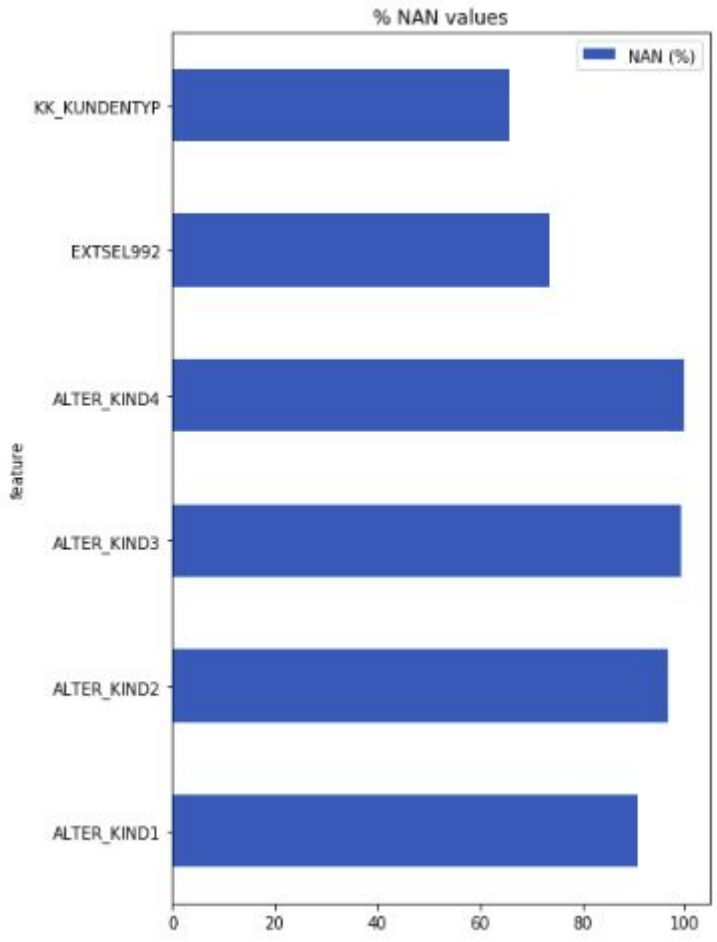


Figure 7.3.3 - Features with more than 60% of their null values from the AZDIAS data set

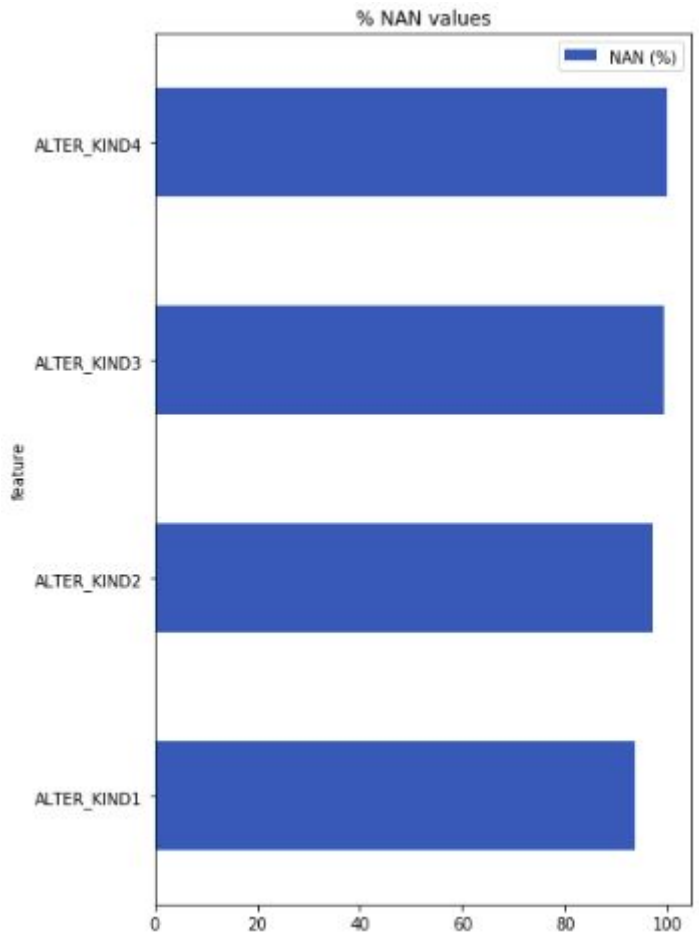


Figure 7.3.4 - Features with more than 60% of their null values from the Consumers data set

After analyzing the 4 graphs, we decided to exclude features that have more than 30% of their null values from both datasets. From this process the features AGER_TYP, ALTER_HH, ALTER_KIND1, ALTER_KIND2, ALTER_KIND3, ALTER_KIND4, EXTSEL992, KBA05_BAUMAX, KK_KUNDENTYP, TITEL_KZ were excluded from the AZDIAS and CONSUMERS data set. In addition, KKK and REGIOTYP have more than 30% of their null values in the CUSTOMERS database, but not in the AZDIAS dataset, with about 17.3% of their null values. We decided that both features were also eliminated in the AZDIAS dataset.

We will analyze which lines should be deleted. For that, we will analyze the number of null values of each line. That is, how many features have null values for a given person. Graph 7.3.5 represents a histogram of the percentage of null values per line of the AZDIAS dataset, and graph 7.3.6 the same histogram, but for the consumers dataset.

Figure 7.3.5 - Histogram of the percentage of null values per line in the AZDIAS dataset

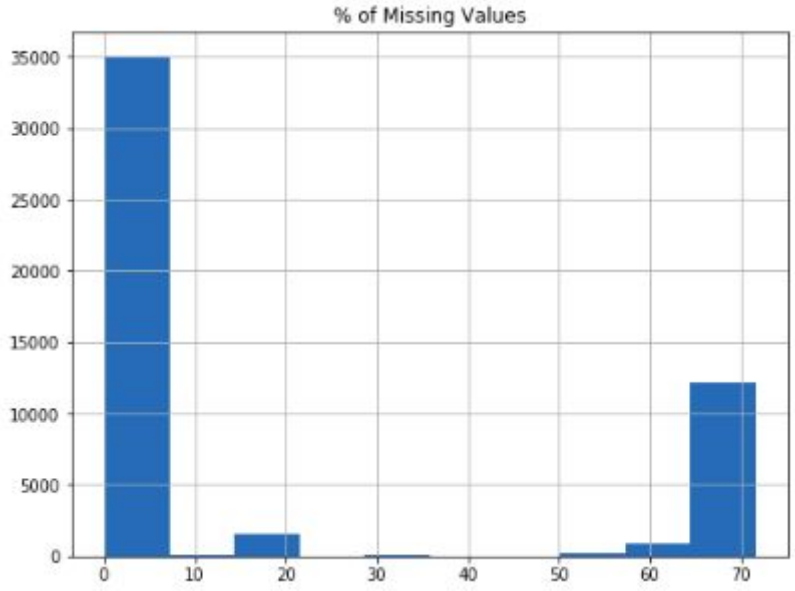


Figure 7.3.6 - Histogram of the percentage of null values per line in the customers dataset

The vast majority of lines have less than 10% of null values. In the customers dataset, a considerable part of the lines have null values between 60% and 70%, as shown in the histogram of figure 7.3.6. The threshold to eliminate lines was chosen as 50%, that is, lines that have more than 50% of null values will be eliminated. Proportionally we will eliminate more lines from the customers dataset.

There are many columns that still do not add new information and that have a high correlation with other features. This can impact the prediction models that we will use in the next

step, because depending on the model, as in prediction trees, highly correlated features negatively impact the result of training the algorithm.

In this way, we will analyze the correlation between features. Table 7.3.7 illustrates some of the features with the strongest negative correlation in AZDIAS data. Table 7.3.8 illustrates in a similar way, but referring to consumers data.

feature 1	feature 2	correlation
GREEN_AVANTGARDE	PRAEGENDE_JUGENDJAHRE_MOVEMENT	-1.000000
GEMEINDETYP	ORTSGR_KLS9	-0.934403
WOHNLAGE_URBAN_OR_RURAL	WOHNLAGE_QUALITY	-0.893610
KOMBIALTER	PRAEGENDE_JUGENDJAHRE_GENERATION	-0.884816
D19_VERSI_ANZ_24	D19_VERSI_DATUM	-0.877056
KBA13_SITZE_4	KBA13_SITZE_5	-0.866568
SEMIO_VERT	ANREDE_KZ	-0.850248
D19_VERSI_ANZ_12	D19_VERSI_DATUM	-0.843546
PRAEGENDE_JUGENDJAHRE_GENERATION	LP_LEBENSPHASE_FEIN_AGE	-0.831403
FINANZ_SPARER	FINANZ_VORSORGER	-0.820745

Table 7.3.7 - Features with more accentuated negative correlation of AZDIAS data

feature 1	feature 2	correlation
GREEN_AVANTGARDE	PRAEGENDE_JUGENDJAHRE_MOVEMENT	-1.000000
GEMEINDETYP	ORTSGR_KLS9	-0.930091
D19_VERSI_ANZ_24	D19_VERSI_DATUM	-0.887009
WOHNLAGE_URBAN_OR_RURAL	WOHNLAGE_QUALITY	-0.882446
KOMBIALTER	PRAEGENDE_JUGENDJAHRE_GENERATION	-0.862922
D19_VERSI_ANZ_12	D19_VERSI_DATUM	-0.853592
FINANZ_SPARER	FINANZ_VORSORGER	-0.849830
SEMIO_VERT	ANREDE_KZ	-0.840371
KBA13_SITZE_4	KBA13_SITZE_5	-0.816459
SEMIO_KULT	ANREDE_KZ	-0.815401

Table 7.3.8 - Features with stronger negative Correlation of Customers data

We realized with this that there are features that have the same meaning, or similar result as GREEN_AVANTGARDE and PRAEGENDE_JUGENDJAHRE_MOVEMENT. Features with a ratio less than -0.95, we will consider as identical features, and therefore, we will eliminate them.

In the next table, we show the 10 features with the highest positive correlation. Table 7.3.9 illustrates some of the features with the strongest positive correlation of AZDIAS data. Table 7.3.10 illustrates in a similar way, but referring to customers data.

	feature 1	feature 2	correlation
39	KBA13_ALTERHALTER_61	KBA13_HALTER_66	0.926758
61	LP_STATUS_FEIN	LP_STATUS_GROB	0.935855
59	LP_FAMILIE_FEIN	LP_LEBENSPHASE_GROB	0.938005
44	KBA13_BAUMAX	PLZ8_BAUMAX	0.949090
51	KBA13_KMH_211	KBA13_KMH_250	0.962873
68	CAMEO_INTL_2015_WEALTH	CAMEO_DEUG_2015_WEALTH_STATUS	0.963162
49	KBA13_HHZ	PLZ8_HHZ	0.967888
46	KBA13_GBZ	PLZ8_GBZ	0.978461
0	ANZ_HAUSHALTE_AKTIV	ANZ_STATISTISCHE_HAUSHALTE	0.978899
45	KBA13_FAB_SONSTIGE	KBA13_HERST_SONST	1.000000

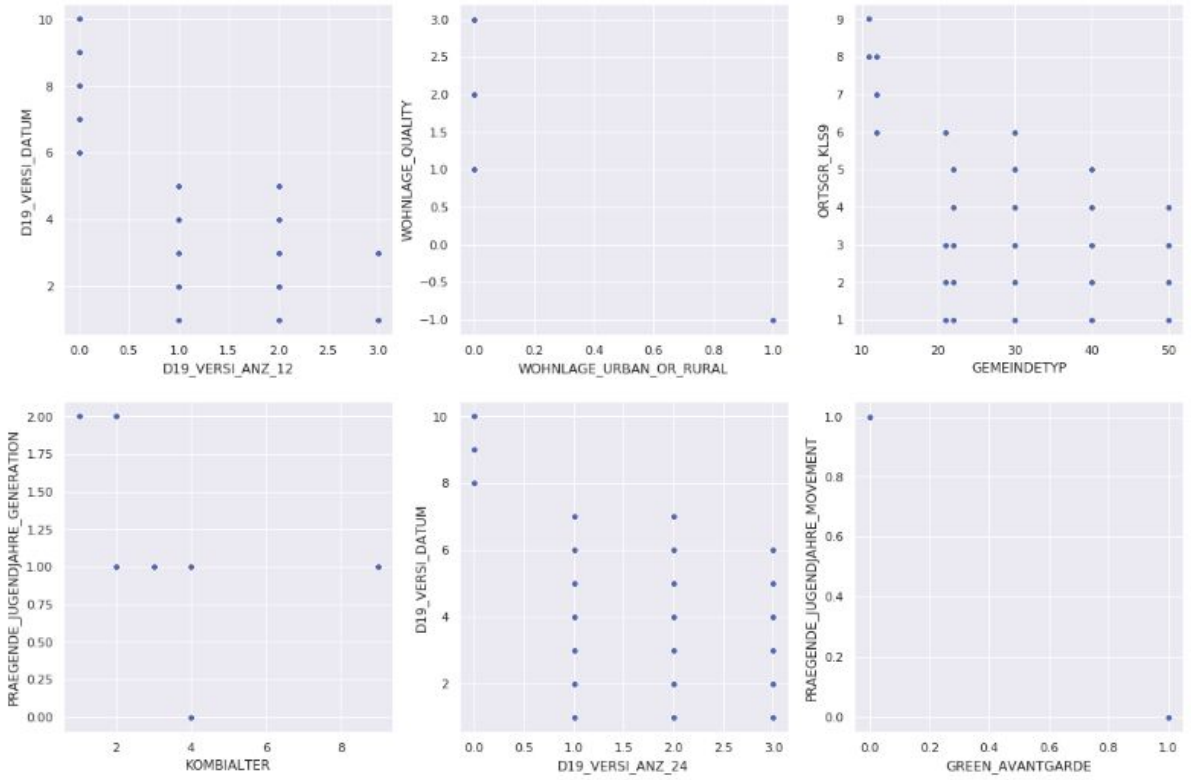
Table 7.3.9 - Features with Stronger Positive Correlation of AZDIAS data

	feature 1	feature 2	correlation
28	KBA13_ALTERHALTER_61	KBA13_HALTER_66	0.929211
47	LP_STATUS_FEIN	LP_STATUS_GROB	0.933257
32	KBA13_BAUMAX	PLZ8_BAUMAX	0.935032
38	KBA13_KMH_211	KBA13_KMH_250	0.962573
54	CAMEO_INTL_2015_WEALTH	CAMEO_DEUG_2015_WEALTH_STATUS	0.963017
36	KBA13_HHZ	PLZ8_HHZ	0.966496
45	LP_FAMILIE_FEIN	LP_LEBENSPHASE_GROB	0.968973
34	KBA13_GBZ	PLZ8_GBZ	0.975931
0	ANZ_HAUSHALTE_AKTIV	ANZ_STATISTISCHE_HAUSHALTE	0.991670
33	KBA13_FAB_SONSTIGE	KBA13_HERST_SONST	1.000000

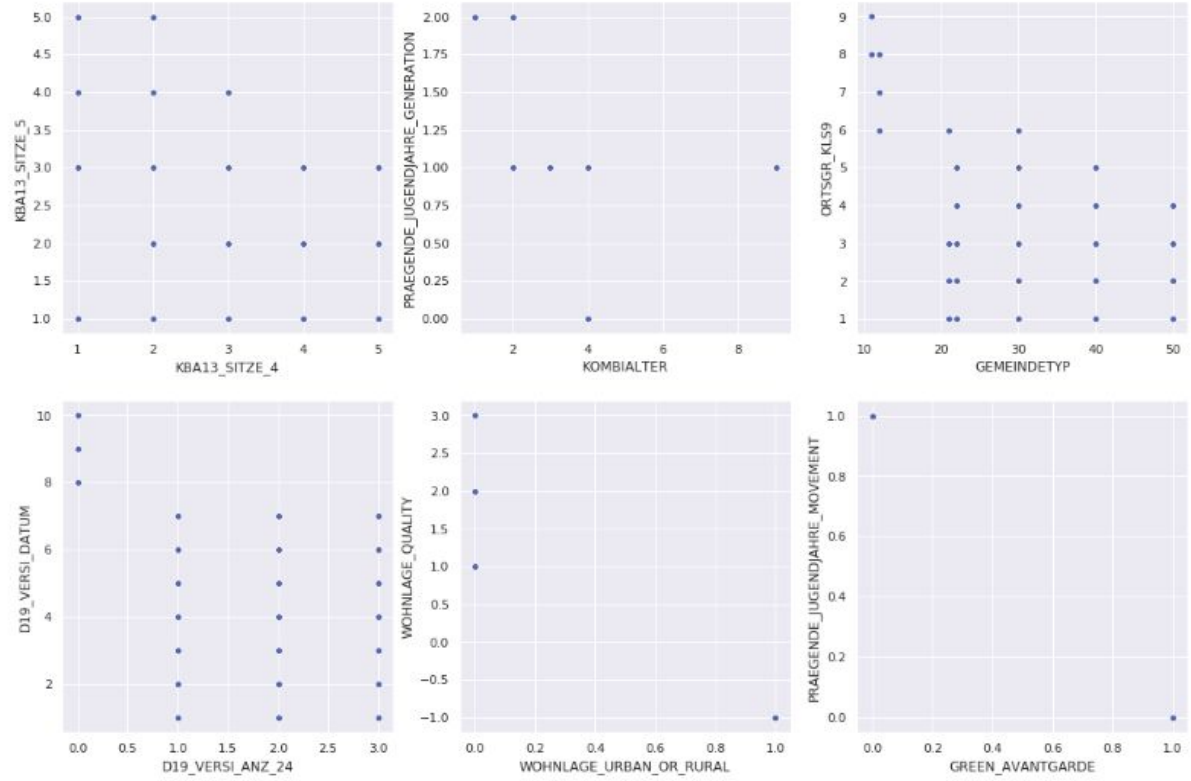
Table 7.3.10 - Features with Stronger Positive Correlation of customers data

We realized with this that there are features that have the same meaning, or similar result as KAB14_FAB_SONSTIGE and KAB13_HERST_SONST. Features with a ratio greater than 0.95, we consider identical features, and therefore, we will eliminate them.

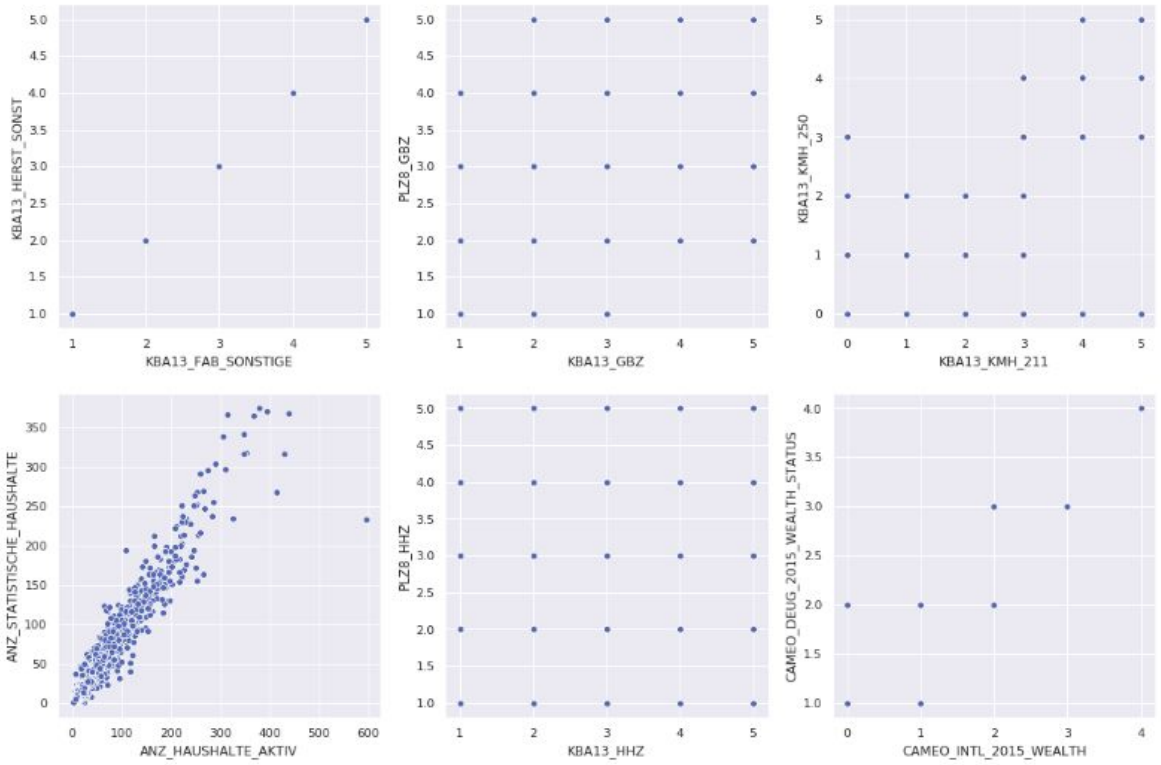
In the next graphs we will plot some of these features with high correlation. The graph 7.3.11 represents the scatterplot the features of AZDIAS with the strongest negative correlation, while the graph 7.3.12 represents the scatterplot the features of customers with the strongest negative correlation. Besides, graph 7.3.13 represents the scatterplot of azdias features with the highest positive correlation, while the 7.3.14 graph represents the scatterplot of customers features with the highest positive correlation.



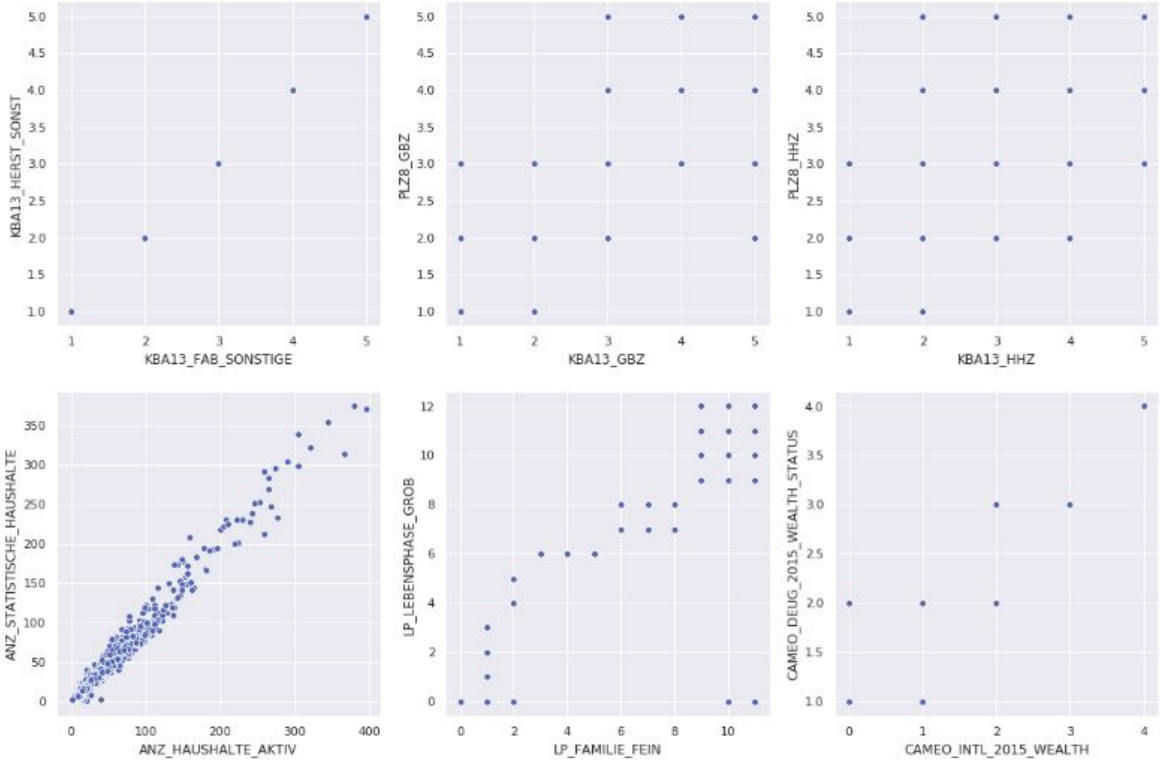
Graph 7.3.11 - Scatter plot of general population features with stronger negative correlation



Graph 7.3.12 - Scatter plot of customers features with stronger negative correlation



Graph 7.3.13 - Scatter plot of general population features with stronger positive correlation



Graph 7.3.14 - Scatter plot of customers features with stronger positive correlation

7.4. Final Processing

The following question is raised: what will we do with the null values of the features? We decided to fill in the null values with the most frequent value for each feature, using the module *SimpleImput* [9].

Finally, we use *StandartScale* module [10] to normalize the values of the features, decreasing them from the average and dividing by the value of the standard deviation of each feature.

8 Dimensionality Reduction and Customer Segmentation Report

We want to deduct the unnecessary features, in order to facilitate our analysis of customer segmentation, in which the rewrite the data by using other variables that are independent of each other and making easier to explore and visualize the dataset. For that, we will use PCA [7].

The graph 8.1 illustrates the number of main components by the percentage of the accumulated variance, in relation to the original data (AZDIAS). The green dashes show the number of main components to maintain 20%, 40%, 60%, 80%, 90%, 95%, 99% and 100% of the original variance.

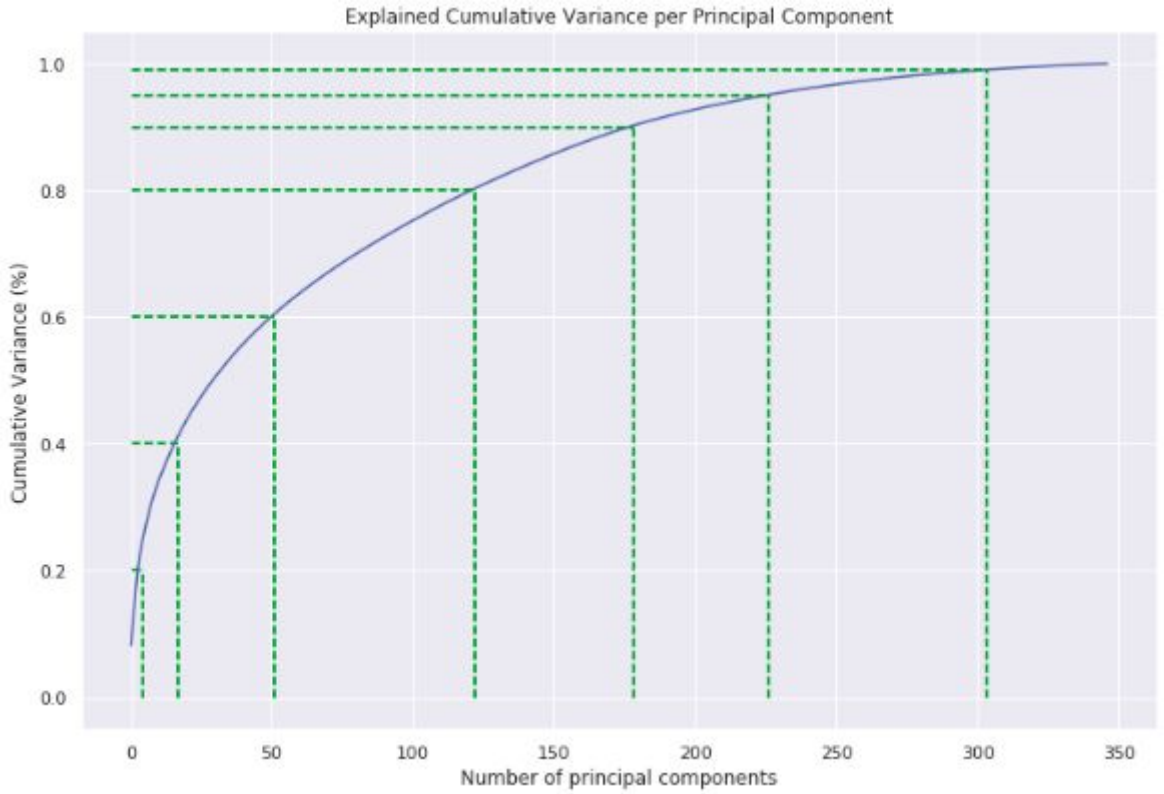


Figure 8.1 - Explained Cumulative Variance per Principal Component

Figure 8.2 shows the features with the highest values (in module) for the zero dimension of principal component analysis.

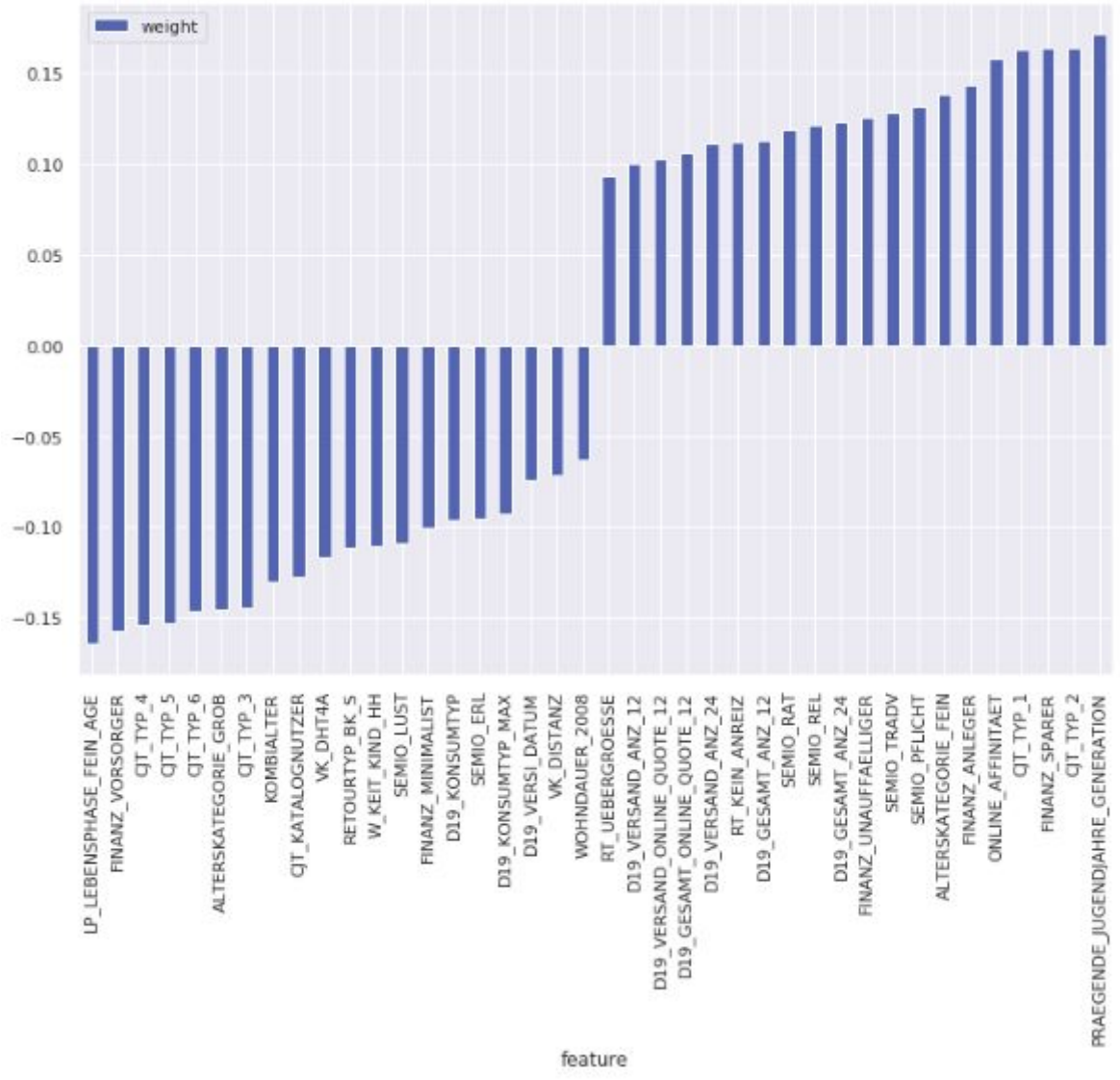


Figure 8.2 - Highest values (in module) for the zero dimension of principal component analysis.

Then, we use the K-means [8] algorithm as a clustering approach to segment our database of the general population, in order to separate them into groups that somehow have characteristics in common. For this, we first use PCA to maintain 95% of the original variance. To represent 95% of the original variance, a minimum of 226 features are required.

Figure 8.3 shows the quadratic error in relation to the distance to each respective center cluster. For this, MiniBatchKMeans [12], randomly selecting 10,000 lines each iteration.

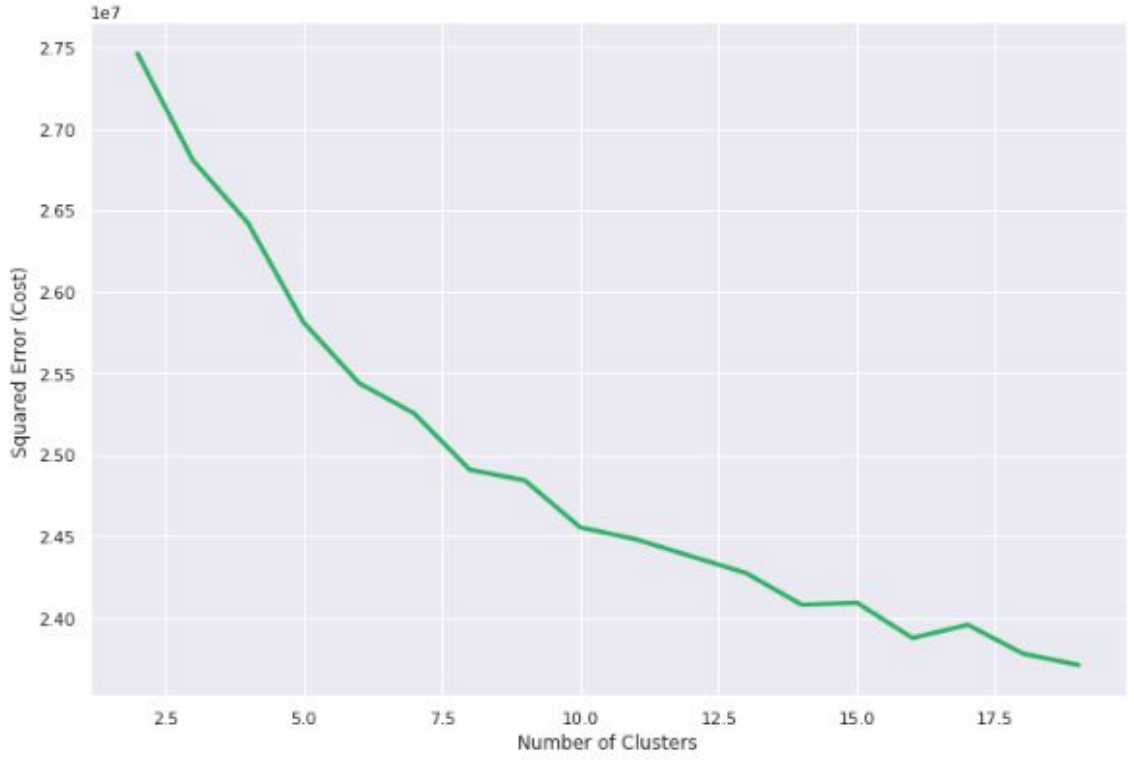


Figure 8.3 - Quadratic Error for Mini Batch K-means

There is no right answer regarding the number of clusters to be considered for segmenting customers. There are some methods like the Elbow method [13], but we will choose by visual inspection the number of clusters equal to 10.

Based on this, we classified our database into 10 clusters numbered from 0 to 9. Figure 8.4 represents the percentage distribution in clusters of the general population and the customer database.

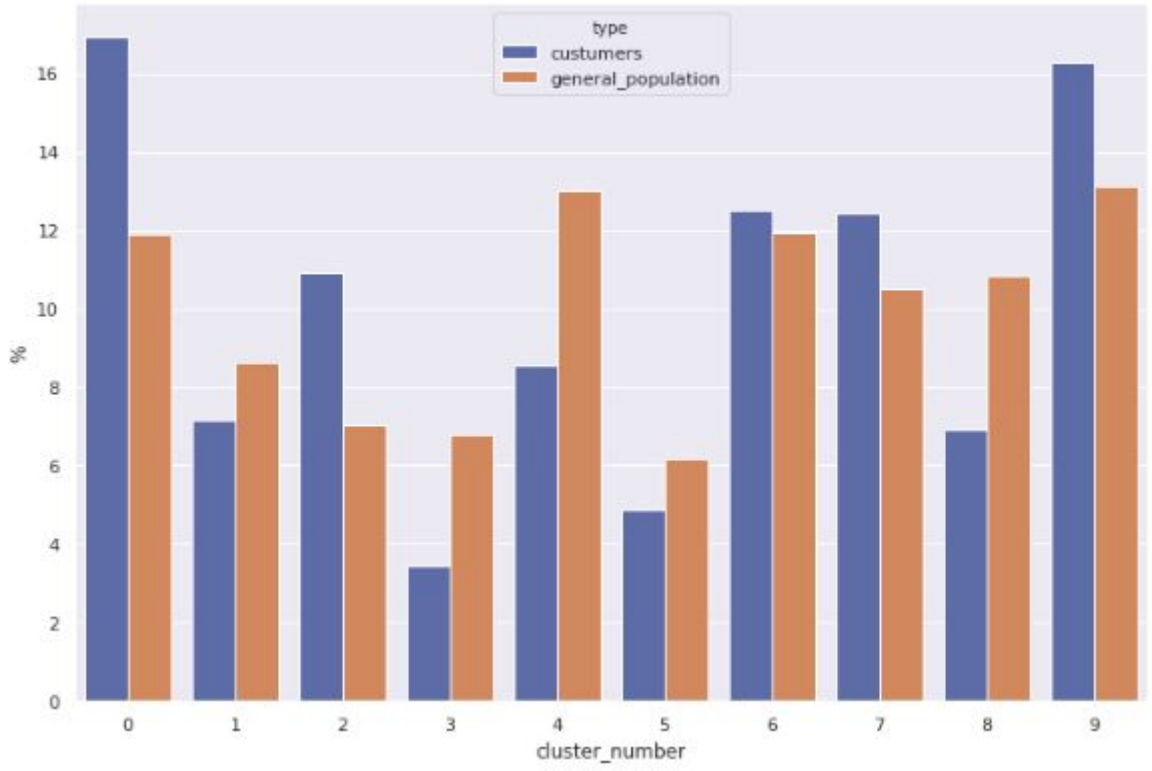


Figure 8.4 - Percentage Distribution between clusters

There is a very similar behavior between the two databases, differing by less than 5% between each of the clusters. Graph 8.5 shows a comparison of the difference between these percentages.

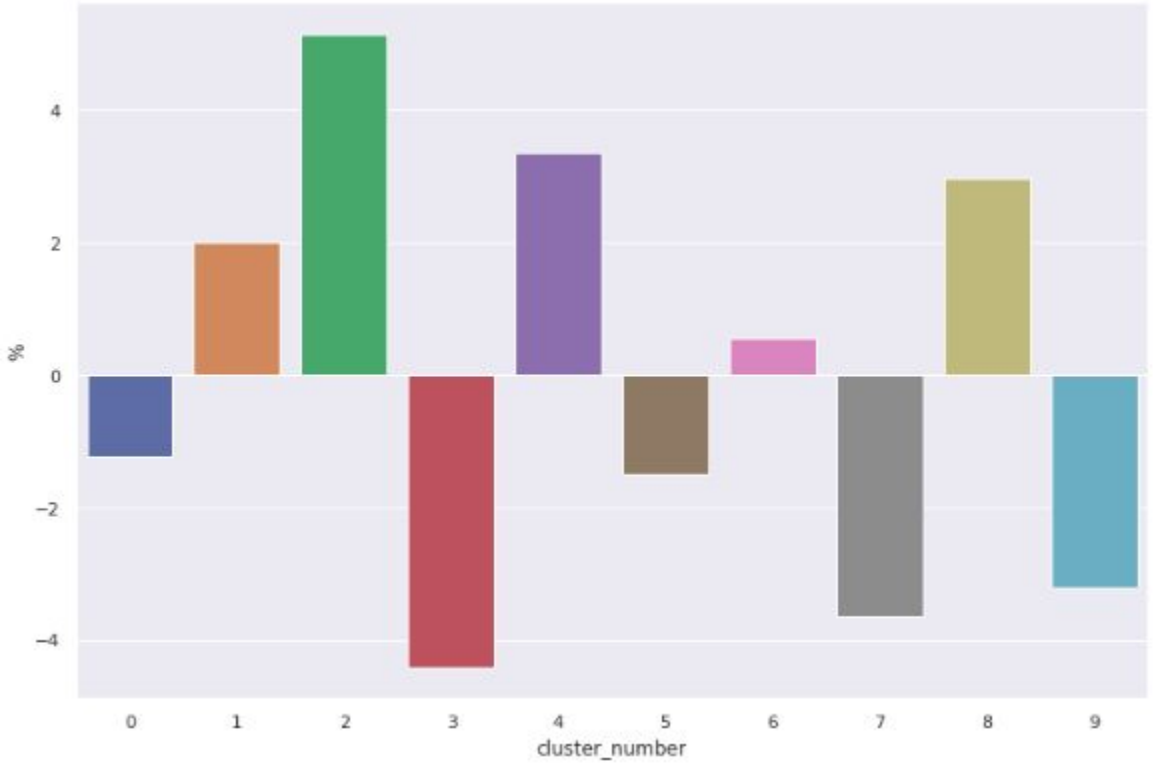


Figure 8.5 - Difference between percentages of general population and customers distribution on clusters.

Below we show a tree map, for each database, relative to the distribution in relation to these clusters. It is clear that the clusters with the highest representativeness are 0 and 9 in the general population, while clusters 4 and 9 have the highest representativeness in the database of the general population.

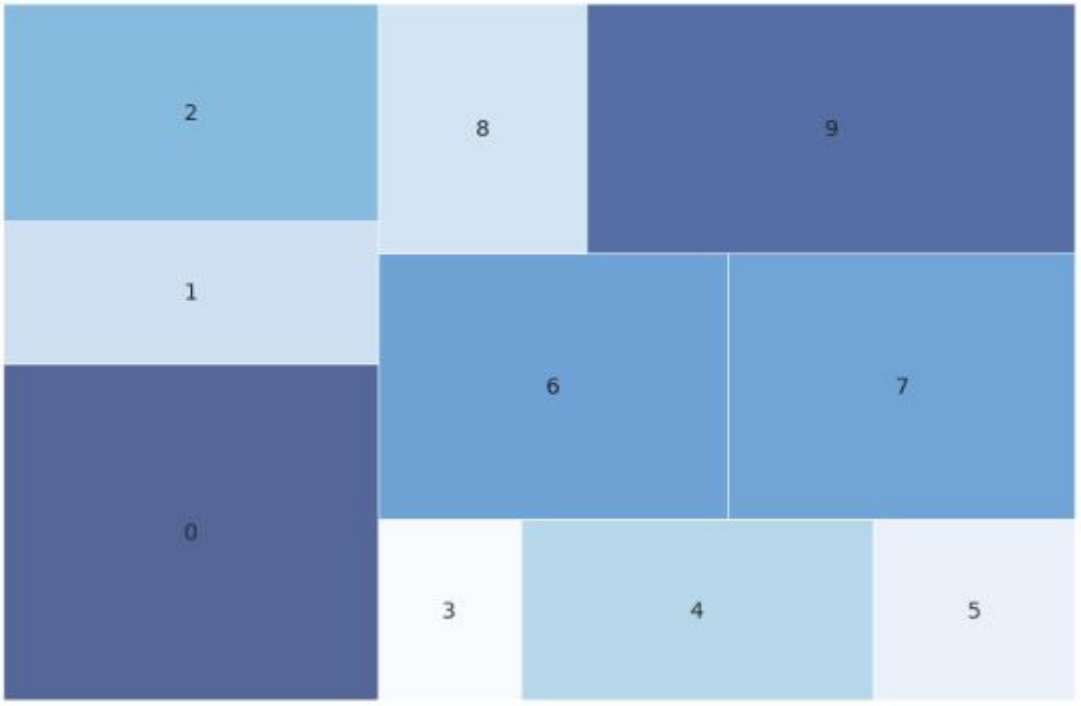


Figure 8.5 - Clusters Distribution in Azdias database

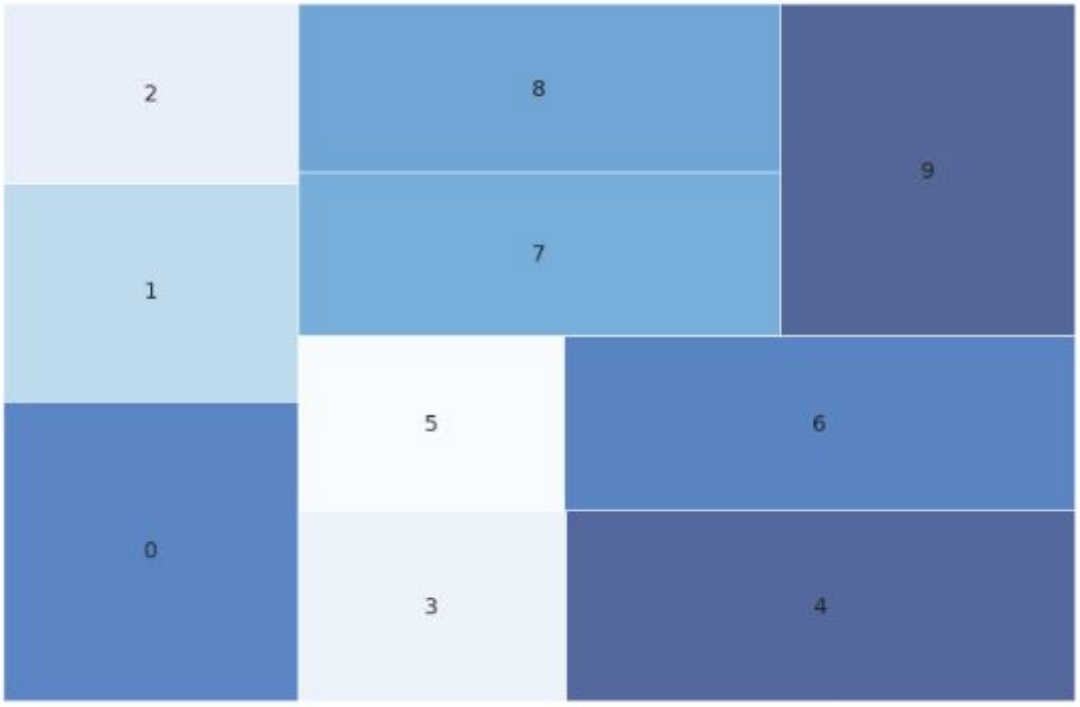


Figure 8.6 - Clusters Distribution in customers database

9. Supervised Learning Model

First, we apply the processing and feature engineering of exploratory data analysis and clustering. The result of the segmentation for the training data is shown in the count plot of figure 9.1 (for the majority class) and 9.2 (for the minority class).

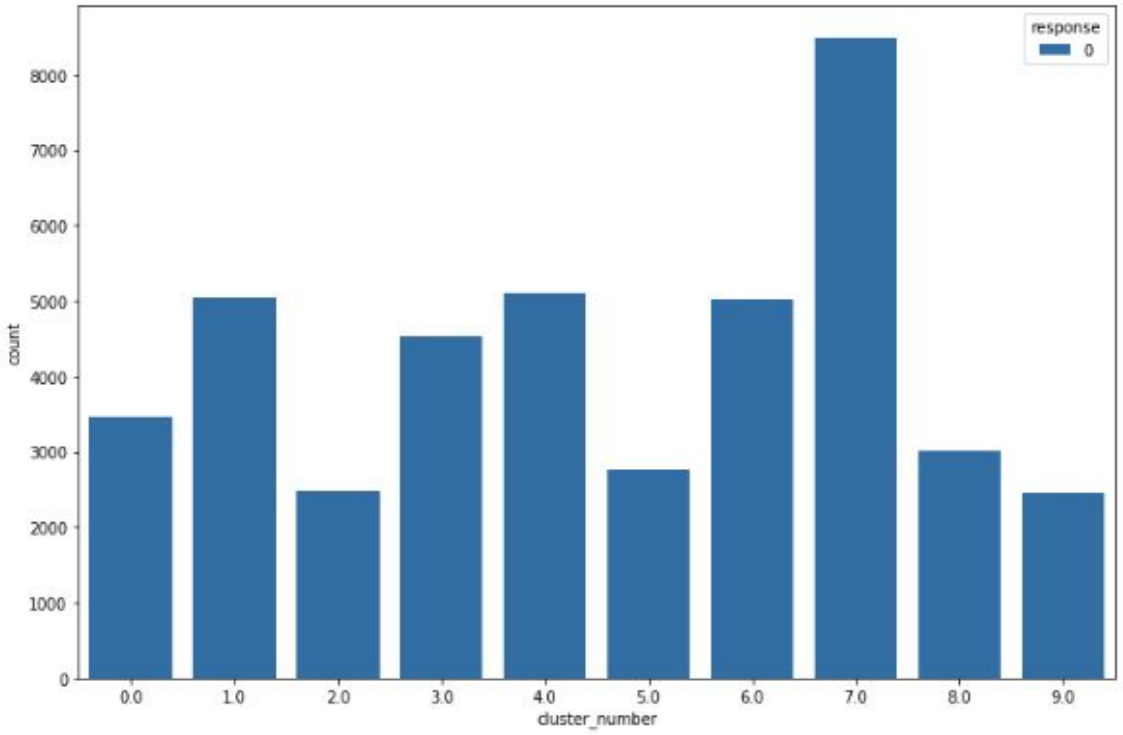


Figure 9.1 - Cluster distribution on training data for RESPONSE = 0, where y-axis is the number of elements in cluster

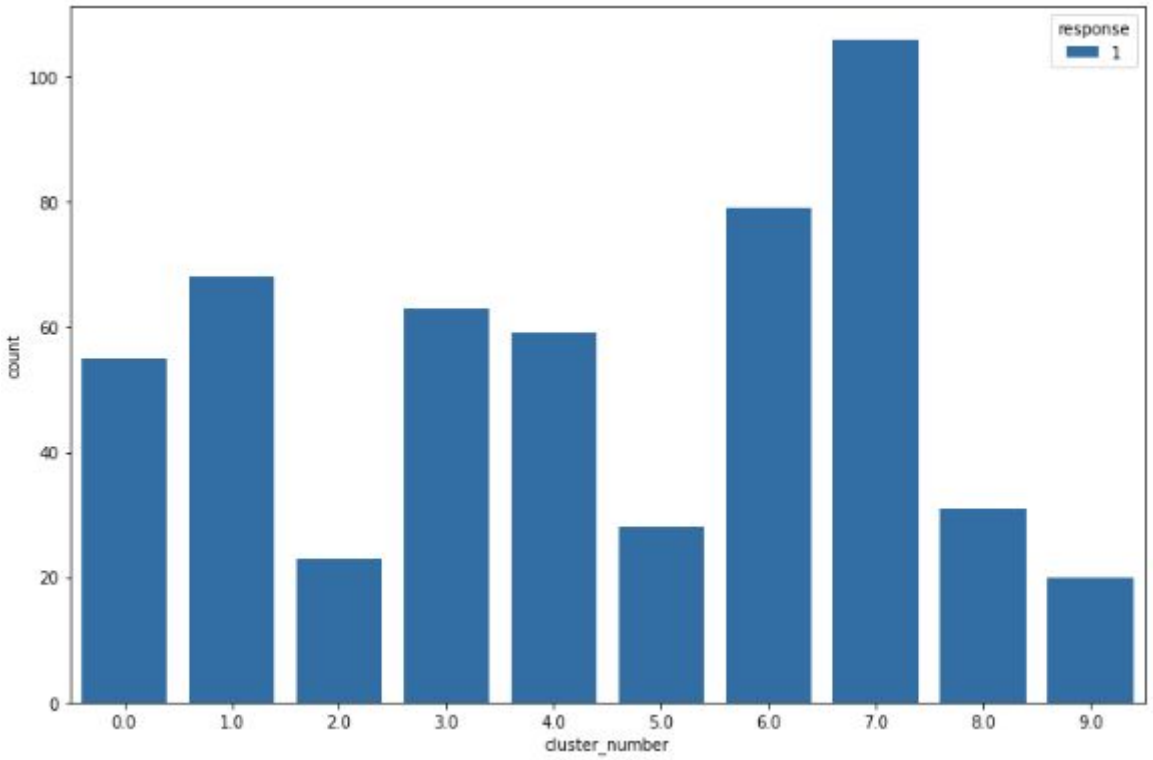


Figure 9.2 - Cluster distribution on training data for RESPONSE = 1, where y-axis is the number of elements in cluster

The behavior is very similar for both classes. With small percentage variations in the distribution between clusters.

We chose 3 models of different nature to try to capture the nature of the data in some way. A logistic model (Logistic Regression) and two ensemble models: a bagging model (Random Forest) and a boosting model (Xgboost). For each model, we use different packages. We use sklearn package for RandomForestClassifier module [6] from sklearn.ensemble and LogisticRegression module from sklearn.linear_model [5]. We use xgboost [4] package for Xgboost classifier.

We will use as baseline models, Logistic Regression, Random Forest and Xgboost with the default parameters and hyperparameters.

There is a major problem of class imbalance, 98.76% for class 0, and 1.24% for class 1. In this way, we will evaluate each of these models by reducing the majority class.

Then, for each of the models, we will use Randomized Grid Search [3] to choose the hyperparameters, using cross-validation with 3 folds, so that each of the 3 folds, the proportion between classes remains the same as the original data. The number of iterations was setted to 300.

We reduced the majority class by 20%, so that the proportion between classes is 98.46% and 1.54%, with little difference between the final proportion. In this way, we will also use the same pipeline in models without downsample (randomized grid search and cross validation with 3 folds for hyper parameter tuning).

The comparative metric to evaluate the models will be the ROC-AUC (Area Under the Curve Receiver Operating Characteristics), explained previously.

Next, we see the choice of parameters used in the Randomized Grid Search process.

```
params_rf = {'bootstrap': [True, False],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400, 600, 800, 1000, 2000],
 'class_weight' :['balanced','balanced_subsample',None]}

clf_rf = RandomForestClassifier()
```

Figure 9.3 - Random Forest algorithm hyperparameters

```
params_lr = {
    'penalty':['l1', 'l2', 'elasticnet'],        # l1 is Lasso, l2 is Ridge
    'solver':['liblinear'],
    'C': np.linspace(0.00002,1,100),
    'class_weight':["balanced",None]
}


clf_lr = LogisticRegression()
```

Figure 9.4 - Hyperparameters of the Logistic Regression algorithm

```
params_xgboost = {
        'min_child_weight': [1, 5, 10],
        'gamma': [0.5, 1, 1.5, 5],
        'subsample': [0.6, 0.8, 1.0],
        'colsample_bytree': [0.01, 0.1, 1.0],
        'max_depth': [3, 10, 30],
        'n_estimators': [200,500,800],
        'max_delta_step':[0,5,10,15,20],
        'scale_pos_weight': [1,40,80]
        }

clf_xgb = xgb.XGBClassifier(
        objective = 'binary:logistic',
        eval_metric = 'auc',
    )
```

Figure 9.5 - Hyperparameters of the XgBoost algorithm

9.1 Comparing results to Benchmark Model

The results for the models in the training set are organized in the following bar graph:

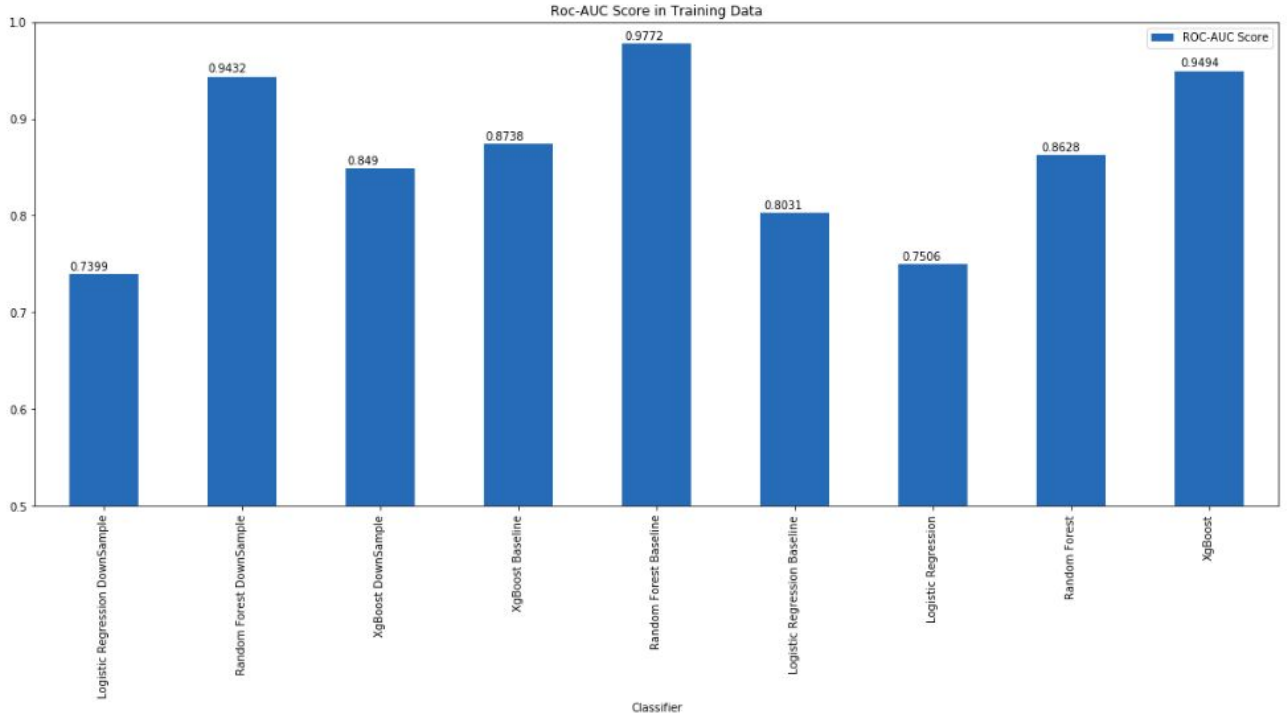


Figure 9.6 - ROC AUC score in Training Data

The model with the highest roc-auc score is the random forest baseline algorithm, however, due to the fact that there is a large imbalance of classes, it is possible for this algorithm to be overfitted. To better evaluate this hypothesis, it is necessary to evaluate the learning curve of each model.

The following 9 graphs show the learning curve of the 9 classifiers we use.

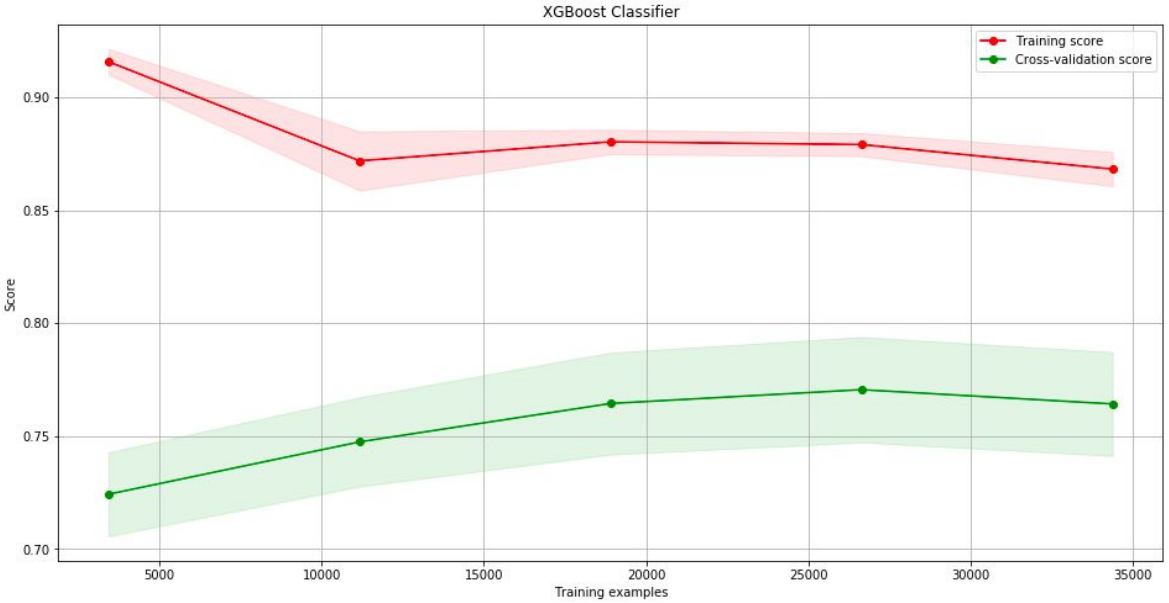


Figure 9.7 - XgBoost Classifier Learning Curve (Roc-AUC score)

There was an improvement in the cross validation score as we increased the number of training examples, and a worsening in the training score. Therefore, we believe that as we increase the number of examples, the model has learned better, reducing overfitting.

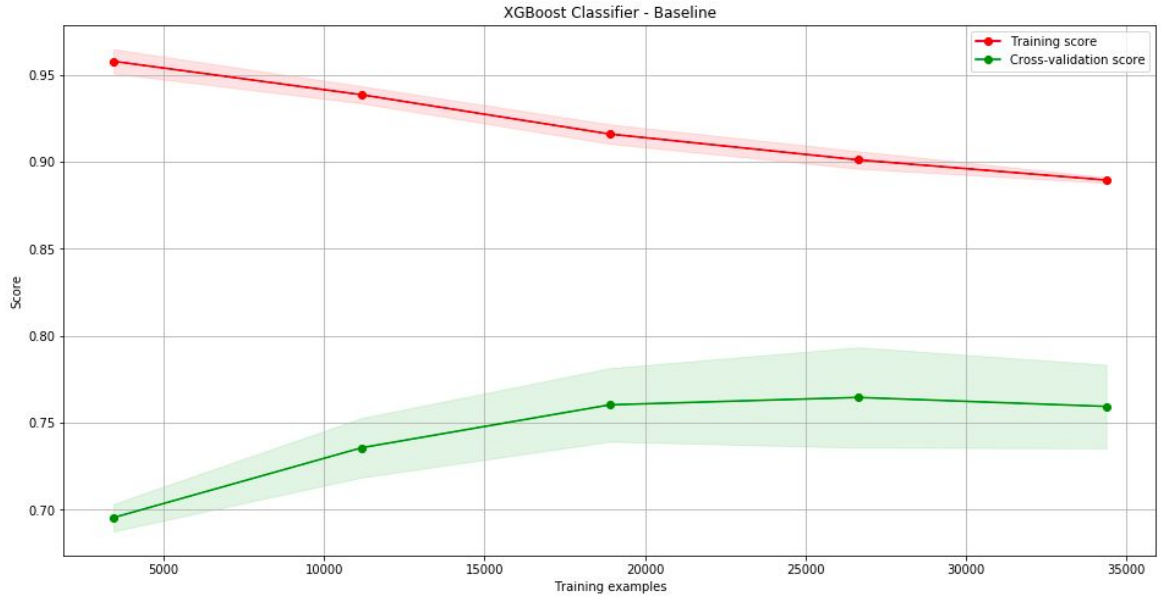


Figure 9.8 - XgBoost Baseline Classifier Learning Curve (Roc-AUC score)

There was a similar behavior in relation to the previous model, with cross-validation score and training score very similar to the previous model. However, a smaller drop in training error is notorious. Thus, I believe that there was greater overfitting in this model, due to the high roc-auc value.

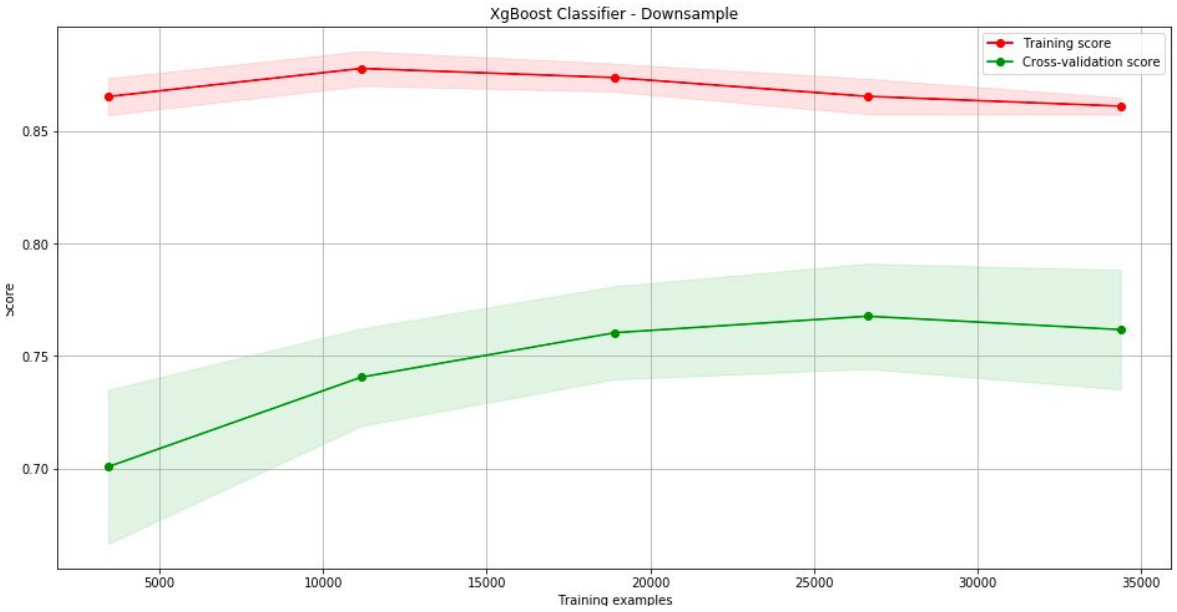


Figure 9.9 - XgBoost Downsample Classifier Learning Curve (Roc-AUC score)

There was an improvement in the cross validation score as we increased the number of training examples, and a worsening in the training score. The training score was lower than the baseline, however the results inf cross validation score were better.

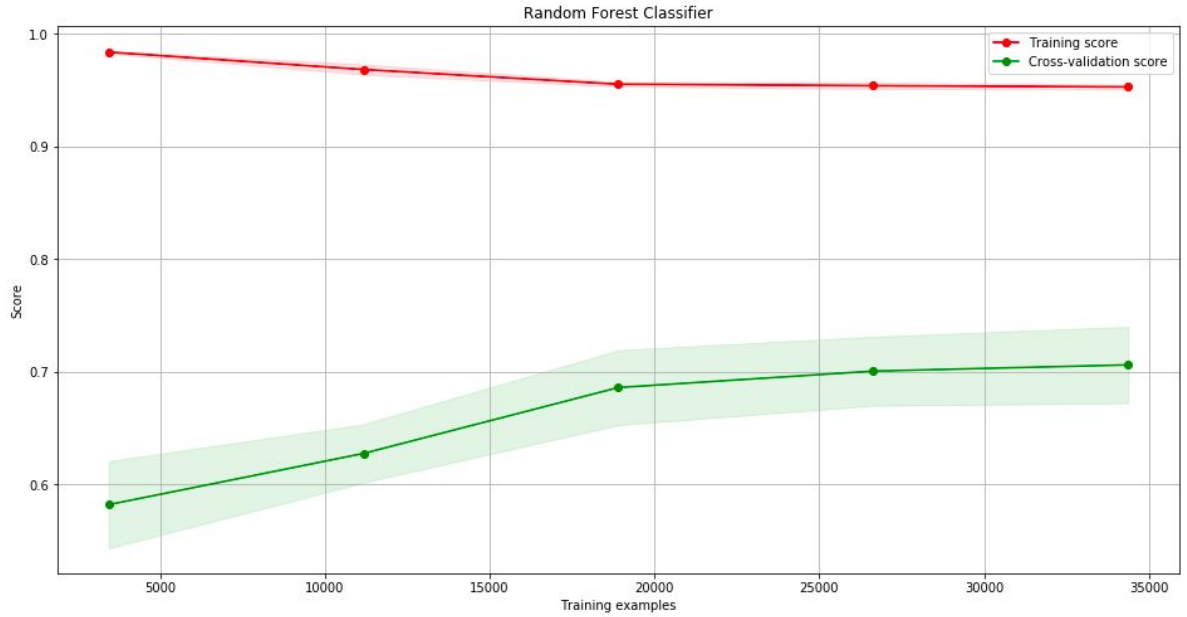


Figure 9.10 - Random Forest Classifier Learning Curve (Roc-AUC score)

The training score hardly changed as we increased the number of training examples. However, the cross-validation score increased as the training set grew, but this score were very lower than the training score. It is quite possible that the model overfitted and is biased (very unbalancing classes).

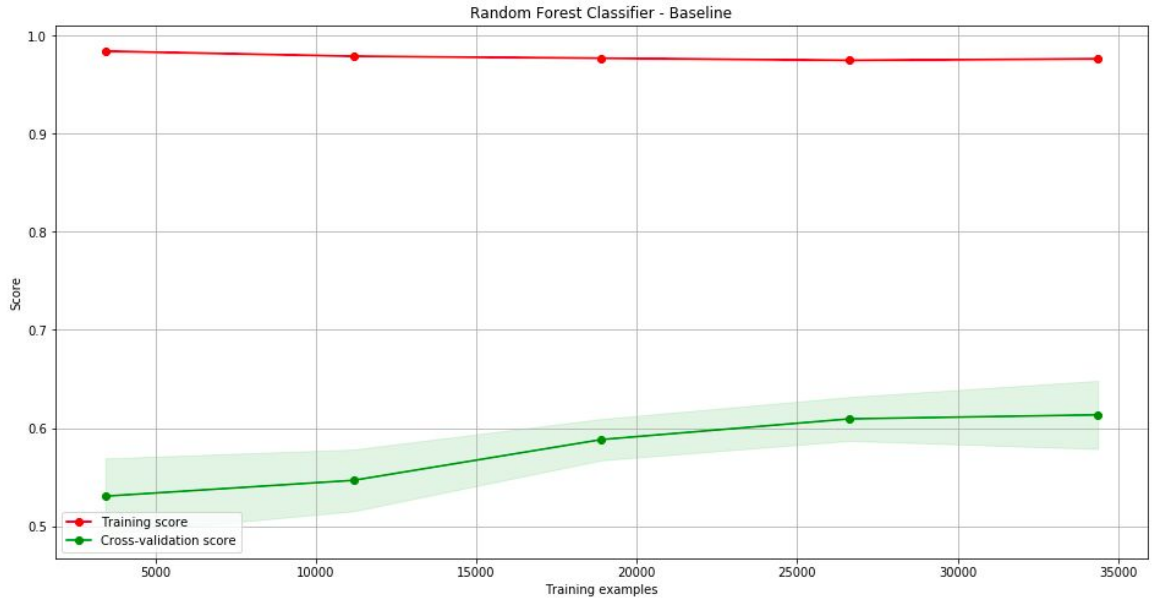


Figure 9.11 - Random Forest Baseline Classifier Learning Curve (Roc-AUC score)

The behavior of the baseline model is very similar to the previous model. It is quite possible that the model overfitted and is biased (very unbalancing classes).

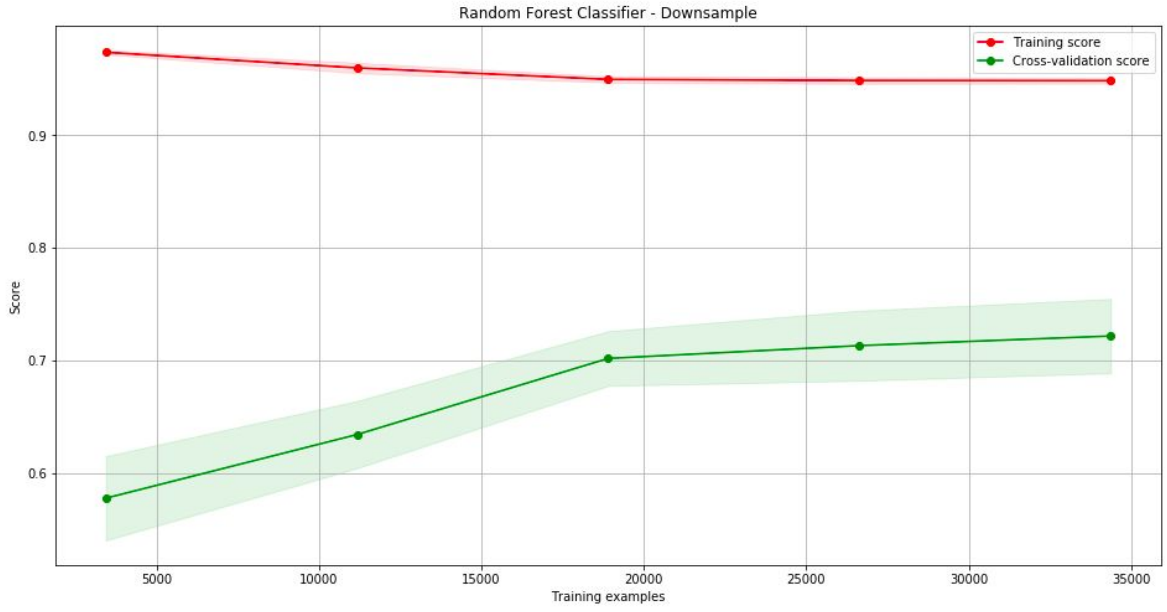


Figure 9.12 - Random Forest Classifier Learning Curve (Roc-AUC score)

The behavior of Downsample model is different from the previous Random Forest classifiers. The cross validation score is higher and the training error is lower, so it is possible that this model will have better performance on unseen data.

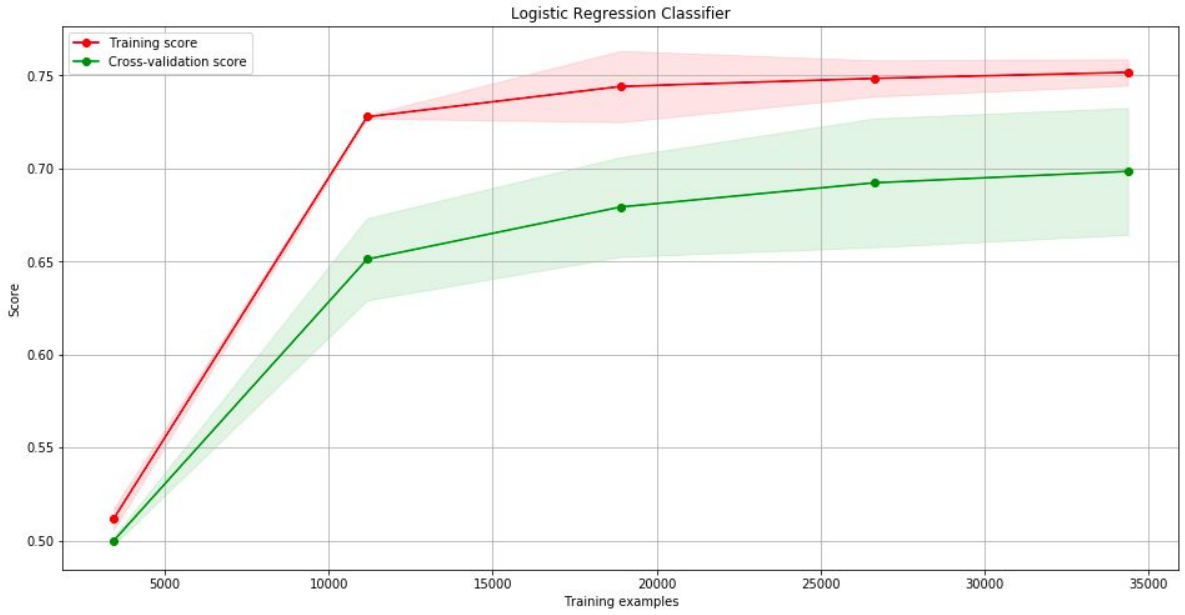


Figure 9.13 - Logistic Regression Classifier Learning Curve (Roc-AUC score)

There was an increase in training error as we increased the size of the dataset. Similar behavior with the cross validation score.

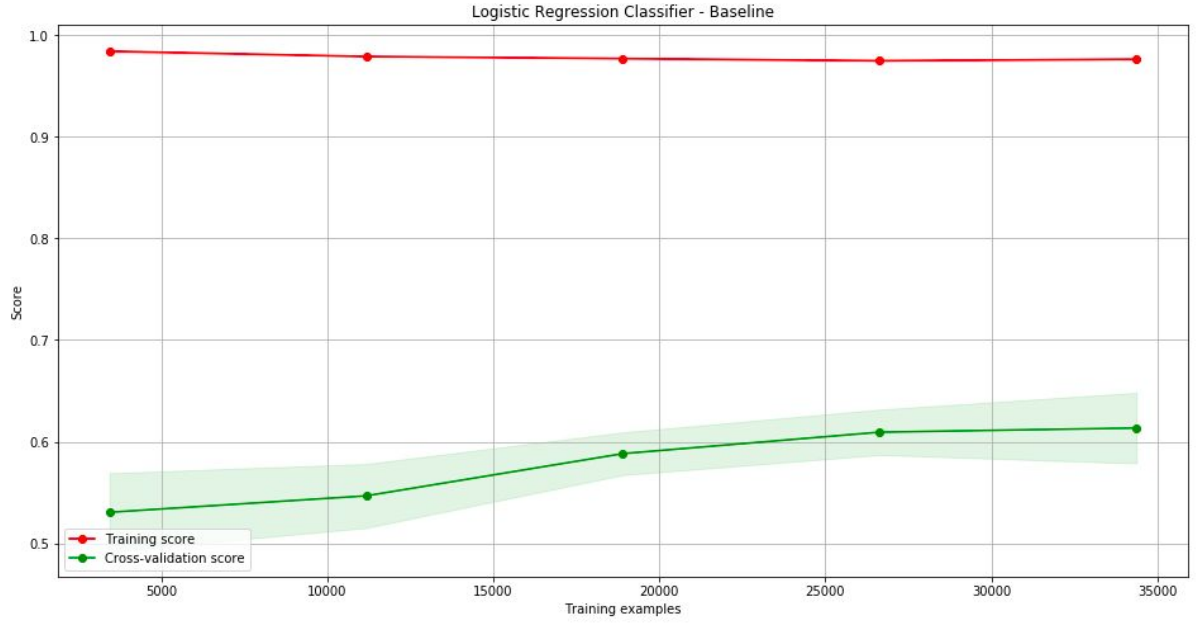


Figure 9.14 - Logistic Regression Baseline Classifier Learning Curve (Roc-AUC score)

There was little change with the increasing number of examples in the training set.

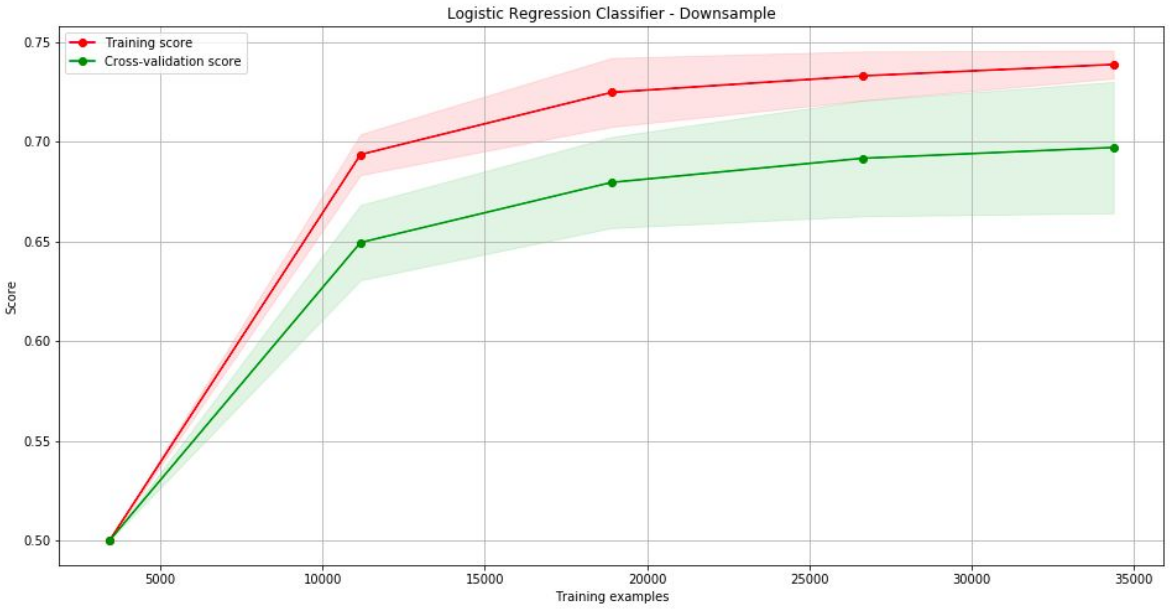


Figure 9.15 - Logistic Regression Downsample classifier Learning Curve (Roc-AUC score)

There was an increase in training error as we increased the size of the dataset. Similar behavior with the cross validation score.

10. Kaggle Submission

The classifier with the best result was the Xgboost downsample with a score of 0.7974. The xgboost models had better results, including the xgboost baseline with the 3rd best result.

All models had better results using the downsample technique. It is possible that if we increase the proportion of the minority class even further, we can obtain better results, since we only increase the proportion of the minority class by 0.3%.

Graph 10.1 shows the results obtained from submitting Kaggle.

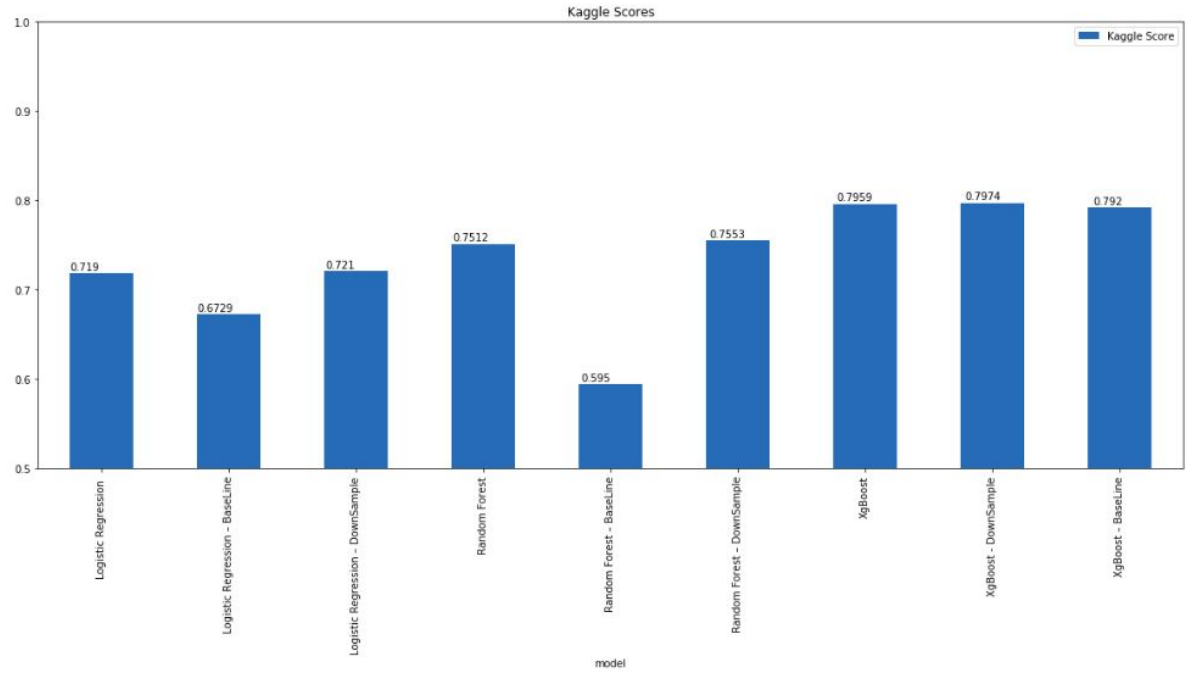


Figure 10.1 - Kaggle Submission Scores

The image 10.2 shows the score of Xgboost algorithm with Downsample and the image 10.3 shows the current position between 179 students.

Figure 10.2 - Score of Xgboost algorithm with Downsample

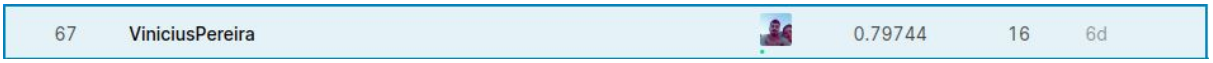
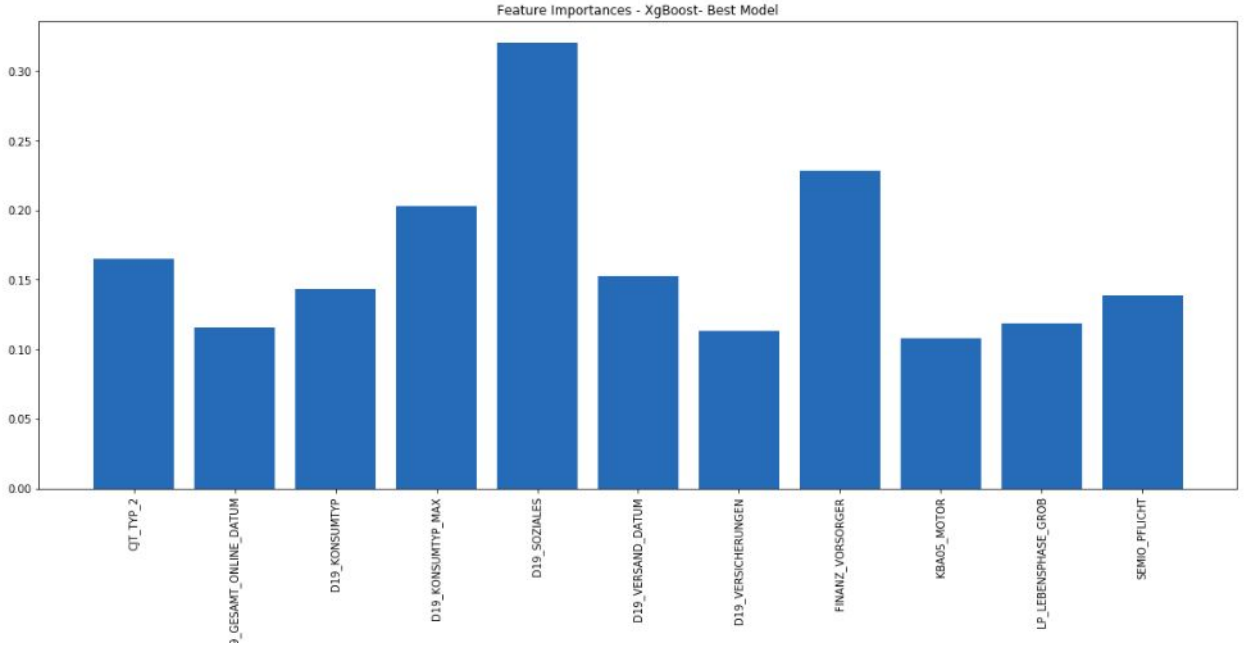


Figure 10.2 - Current position of Xgboost algorithm with Downsample

10.1 Feature Importances

For the xgboost with downsample algorithm, we plot the features with the heaviest weight on the model. The heaviest feature is D19_SOZIALES. Not much information is known about this feature because it is not in the dictionary.



11. Future Working

As a future work, I believe that using GridSearch [14] instead of RandomizedGridSearch with only 300 iterations so that all the possibilities of parameters helps to find a slightly better model. The time taken to evaluate each model was relatively high, which meant that the choice of RandomizedGridSearch was made.

However, what proved to be a good modeling strategy would be to increase the downsample of the majority class, since it obtained better results in the 3 selected algorithms. Thus, I would choose to reduce the majority class further, and increase the number of iterations in RandomizedGridSearch or use GridSearch on a faster computer.

12. References

[1] Kaggle.com. (2020). *Udacity+Arvato: Identify Customer Segments | Kaggle*. [online] Available at: https://www.kaggle.com/c/udacity-arvato-identify-customers/leaderboard [Accessed 1 Mar. 2020].

[2] En.wikipedia.org. (2020). *Receiver operating characteristic*. [online] Available at: https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve [Accessed 1 Mar. 2020].

[3] Scikit-learn.org. 2020. *Sklearn.Model_Selection.Randomizedsearchcv — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html> [Accessed 18 May 2020].

[4] Xgboost.readthedocs.io. 2020. *Python Package Introduction — Xgboost 1.1.0-SNAPSHOT Documentation*. [online] Available at: <https://xgboost.readthedocs.io/en/latest/python/python_intro.html> [Accessed 18 May 2020].

[5] Scikit-learn.org. 2020. *3.2.4.3.1. Sklearn.Ensemble.Randomforestclassifier — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 18 May 2020].

[6] Scikit-learn.org. 2020. *Sklearn.Linear_Model.Logisticregression — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html> [Accessed 18 May 2020].

[7] Scikit-learn.org. 2020. *3.2.4.3.1. Sklearn.Ensemble.Randomforestclassifier — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 18 May 2020].

[8] Scikit-learn.org. 2020. *3.2.4.3.1. Sklearn.Ensemble.Randomforestclassifier — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 18 May 2020].

[9] Pythonhosted.org. 2020. *Welcome To Simpleinput'S Documentation! — Simpleinput V0.1.1 Documentation*. [online] Available at: <https://pythonhosted.org/simpleinput/> [Accessed 18 May 2020].

[10] Scikit-learn.org. 2020. *Sklearn.Preprocessing.Standardscaler — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html> [Accessed 18 May 2020].

[12] Scikit-learn.org. 2020. *Sklearn.Cluster.Minibatchkmeans — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html> [Accessed 18 May 2020].

[13] En.wikipedia.org. (2020). *Elbow method (clustering)* [online] Available at: https://en.wikipedia.org/wiki/Elbow_method_(clustering) [Accessed 1 Mar. 2020].

[14] Scikit-learn.org. 2020. *Sklearn.Model_Selection.Gridsearchcv — Scikit-Learn 0.23.0 Documentation*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html> [Accessed 18 May 2020].