February 24, 2021

1 Project Definition

The purpose of the following report is to explain the approaches taken to solve the Arvato project, Customer Segmantation which is the final part of the Machine Learning Nanodegree Program offered by Udacity. The project was mainly based on two different tasks,

1. Analyzing demographics data of the general population in Germany and identifying potential customers by utilizing the demographics data in their current customer base. This was modeled as an unsupervised task.
2. Given a dataset with labeled data that describes if an individual has responded to a mail-out or not, try and predict which individuals are most likely to become a customer (by responding to the mail-out).

In order to tackle the problems above, the following approaches were taken,

1. **Data exploration and cleaning** which included data visualization, handling missing values, removing columns and rows, encoding categorical variables and standardizing the input data.
2. **Utilization of dimensionality reduction technique**, principal component analysis (PCA) to reduce the dimension of the input data.
3. K-means algorithm to create different clusters based on the general population dataset and used it to assign existing customers to a specific cluster.
4. Finally, developed and compared multiple classification models in order to select the best one (based on ROC), which in this case was Logstic regression.

2 Data description

In total, 6 datasets where given in an excel/csv format. A more in depth description for each dataset can be seen below.

- **DIAS Attributes - Values 2017.xlsx** - Mapping of each column in the csv files, including description and the unique values they take.
- **DIAS Information Levels - Attributes 2017.xlsx** - Description of the columns in the csv files broken down into different categories such as person, building etc.
- **Udacity_AZDIAS_052018.csv** - Demographics data for the general population of Germany of the size 891211 persons (rows) x 366 features (columns).
- **Udacity_CUSTOMERS_052018.csv** - Demographics data for customers of a mail-order company of the size 191,652 persons (rows) x 369 features (columns)
- **Udacity_MAILOUT_052018_TRAIN.csv** - Demographics data for individuals who were targets of a marketing campaign of the size 42,982 persons (rows) x 367 (columns).
- **Udacity_MAILOUT_052018_TEST.csv** - Demographics data for individuals who were targets of a marketing campaign of the size 42,833 persons (rows) x 366 (columns).

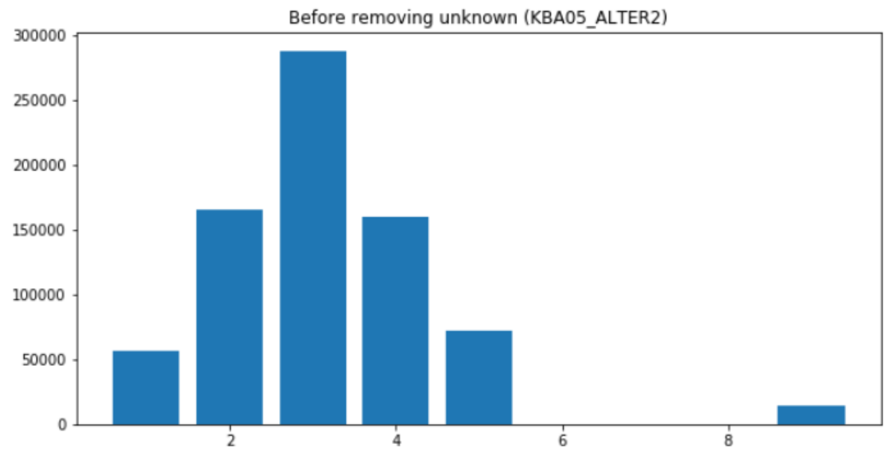
3 Methodology & Results

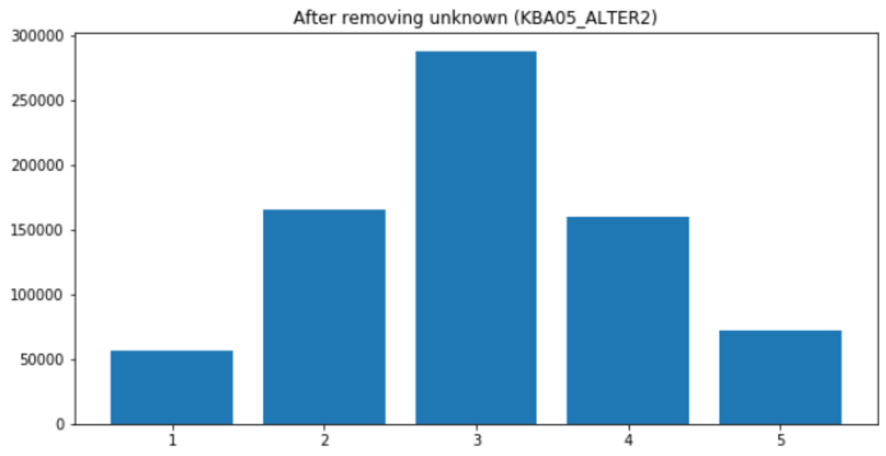
In this section, the methodologies applied to solve each task as well as the results is presented.

3.1 Data exploration and preprocessing

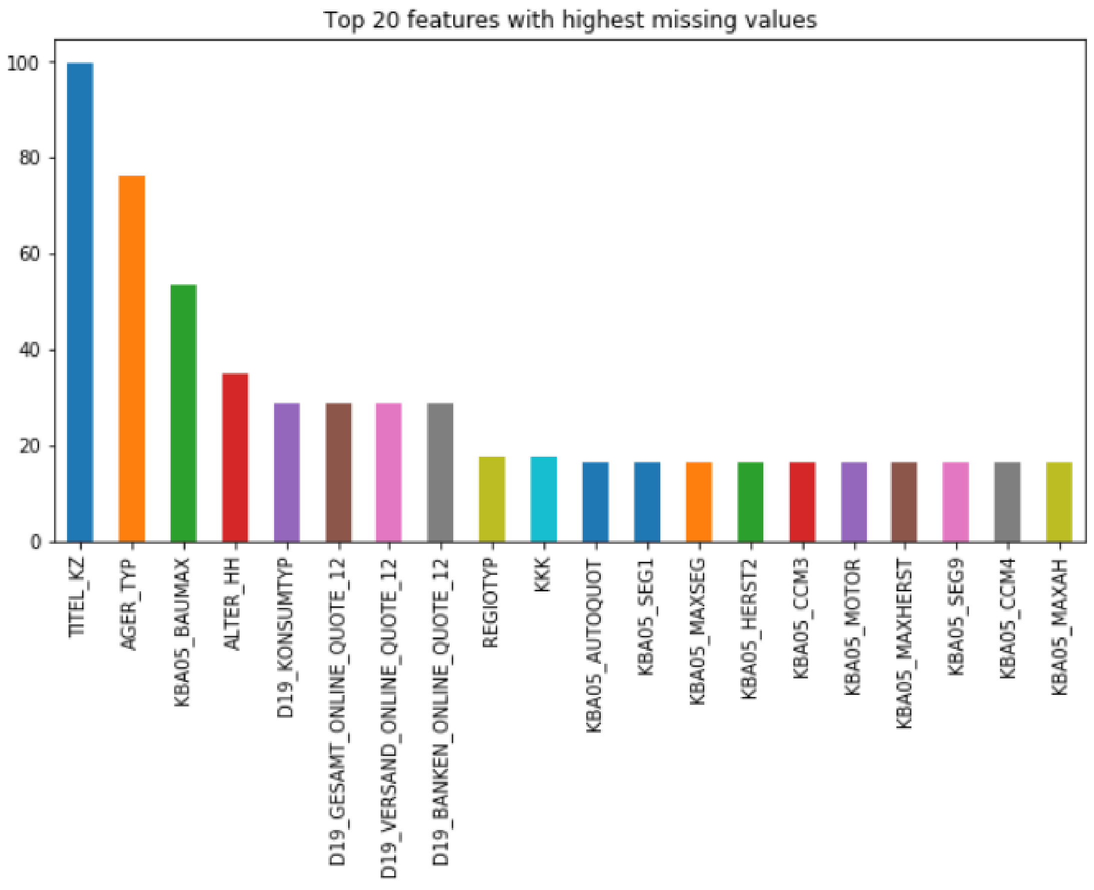
To begin with, a comparison between the columns found in the csv files (i.e. the ones containing individual data) and the excel files which contained the description of each attribute was made. It was noticed that only 272 attributes had an explanation. Therefore, it was decided to remove the attribute/columns that didn't have a description due to simplicity as well as facilitating the data pre-processing (e.g. it is hard to encode a variable where one does not know the type). By performing this step, the number of columns was reduced from 366 to into 272 in the AZIDAS dataset.

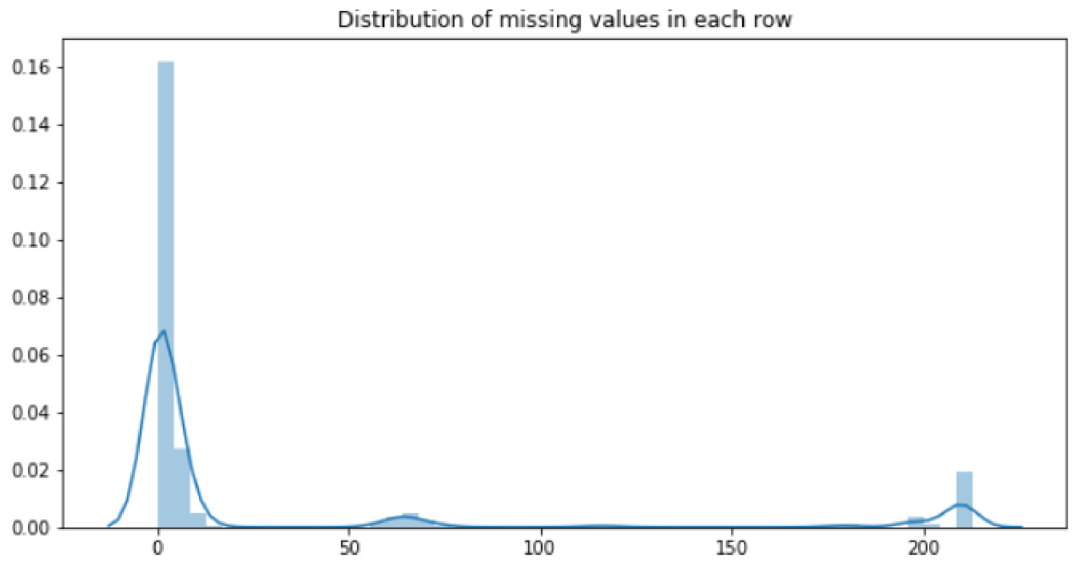
Moving on, the missing values in both the columns and rows in the AZDIAS dataset was investigated. However, a small adjustment was made before identifying the missing values, this involved replacing "unknown"/"no transaction known" (encoded as -1,0 or 9 depending on the attribute) values with NaN. The figures below showcase an example of an attributes distribution before and after replacing the "unknown" (encoded as 9) with NaN.





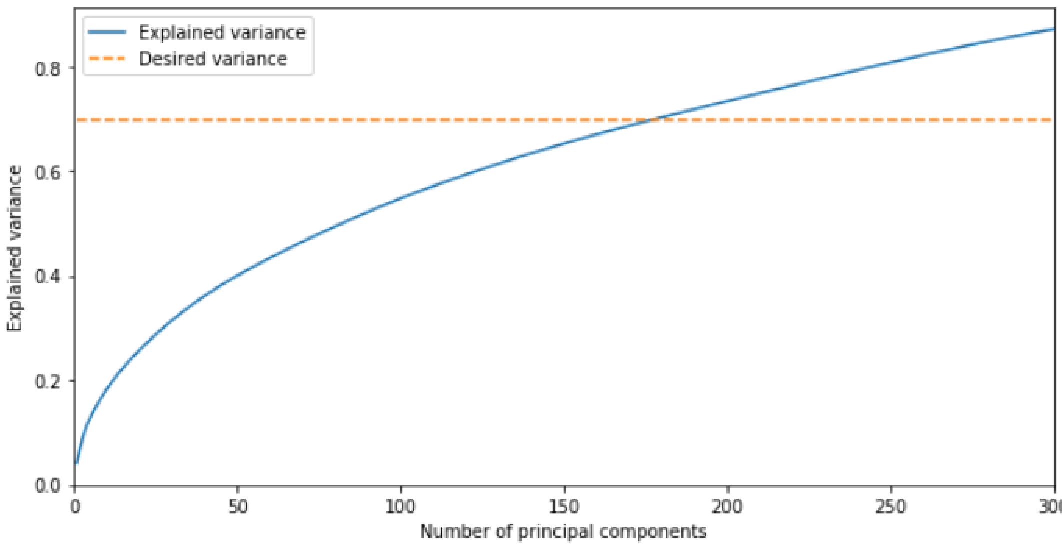
It was then decided that all the columns that had more than 50% missing values as well as rows with more than 50 missing values were dropped. As a result the AZDIAS dataset was reduced into (737288, 269), (rows x columns).





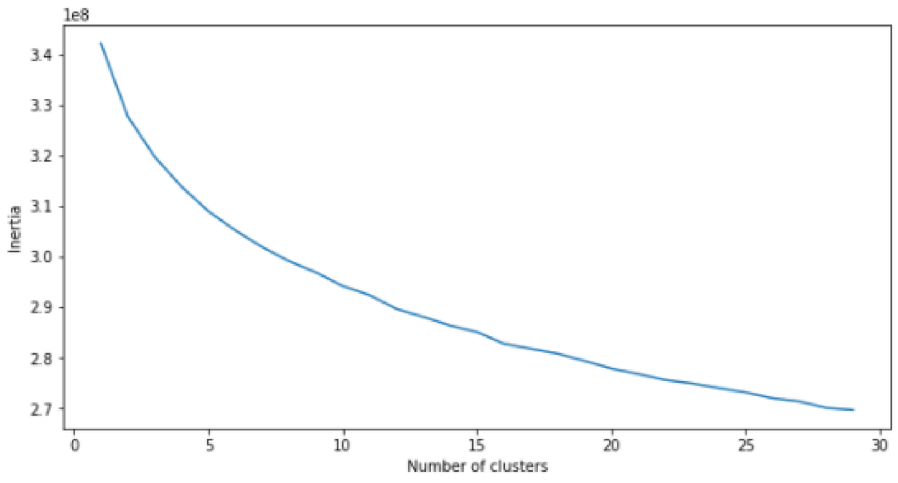
In order to perform Kmeans clustering, handling missing values is necessary. This was done by imputing -1 into the numerical columns and "Unknown" into the string columns. After handling the missing values, it was important to encode the features correctly. The approach taken was to encode nominal features using one-hot-encoding while leaving the other parameters as it was. Unfortunately, the column type was not given, i.e. nominal, ordinal etc. and therefore, a manual investigation of each feature had to be done in order to categorize them.

The final step in the preprocessing was to reduce the dimension of the input data. This was achieved by applying principal component (PCA) which can be seen as constructing new axis where most of the variance is captured. An important step before applying PCA is to standardize the input data. Furthermore, choosing the number of components was done by setting a threshold on the desired variance to be capture, which in this case was set to 70%. As a result of all the steps taken, the input data ended up to be (737288 x 177), (rows x columns).



3.2 Unsupervised learning

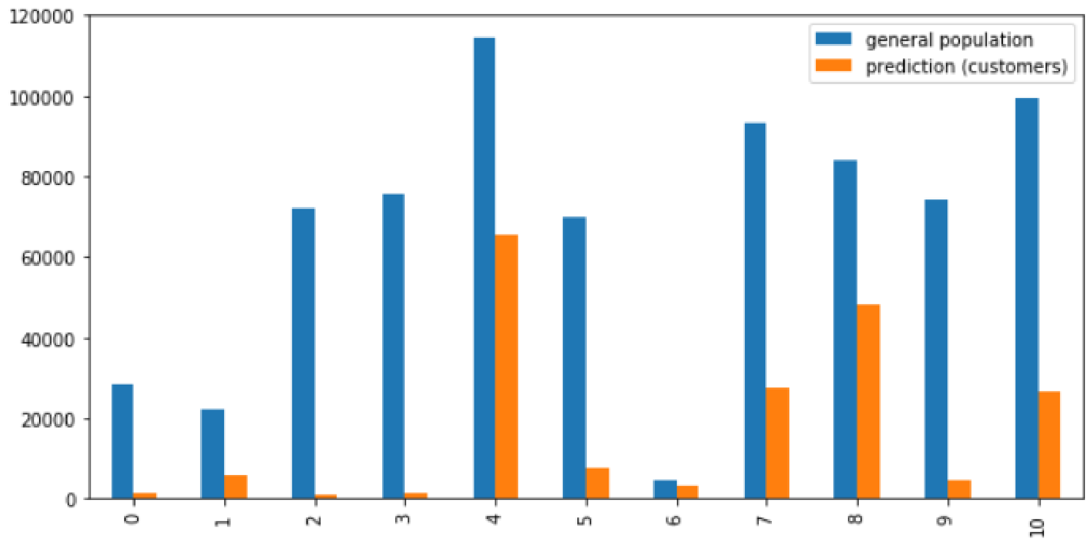
After pre-processing the AZDIAS dataset, Kmeans was applied. One of the main challenges in Kmeans clustering is to find the optimal number of clusters. This is often done by plotting the inertia which measures how internally coherent clusters are, against the number of clusters, and hopefully an elbow shaped graph will be obtained. Thus, the optimal number of clusters will correspond to the "middle" of the elbow. In reality, its rare that one is able to identify an unique number of cluster that is seen as "optimal". For instance, looking at the figure below, it is hard to identify a clear cutoff since the graph is not completely shaped as an elbow, thus, everything between 5-15 could be interpreted as the "optimal value".



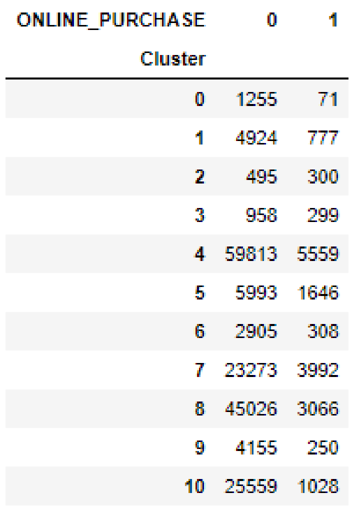
In our case, the number of clusters was set to 11 because its within the "optimal" number of clusters. In addition, I made one assumption that there are 10 clusters/groups in our customer dataset as shown in the table below, and added another class (i.e the 11th) which can be interpreted as individuals that should not be "similar" to our customers.

PRODUCT_GROUP	CUSTOMER_GROUP	ONLINE_PURCHASE
COSMETIC	MULTI_BUYER	0
COSMETIC	MULTI_BUYER	1
COSMETIC	SINGLE_BUYER	0
COSMETIC	SINGLE_BUYER	1
COSMETIC_AND_FOOD	MULTI_BUYER	0
COSMETIC_AND_FOOD	MULTI_BUYER	1
FOOD	MULTI_BUYER	0
FOOD	MULTI_BUYER	1
FOOD	SINGLE_BUYER	0
FOOD	SINGLE_BUYER	1

Having created those 11 clusters, the next step was to assign each customer to a cluster.



As one might notice, the majority of the customers ends up in either cluster 4 or 8. However, there are several aspects one has to consider from a business perspective. Depending on what customer segments one wants to expand within, various decision can be made. From the table below we can see that most of our customers who purchases online is in cluster 4 and 7 and therefore, if a company is trying to acquire customers who likes to purchase online, the focus should be on cluster 4 and 7 rather than 4 and 8.



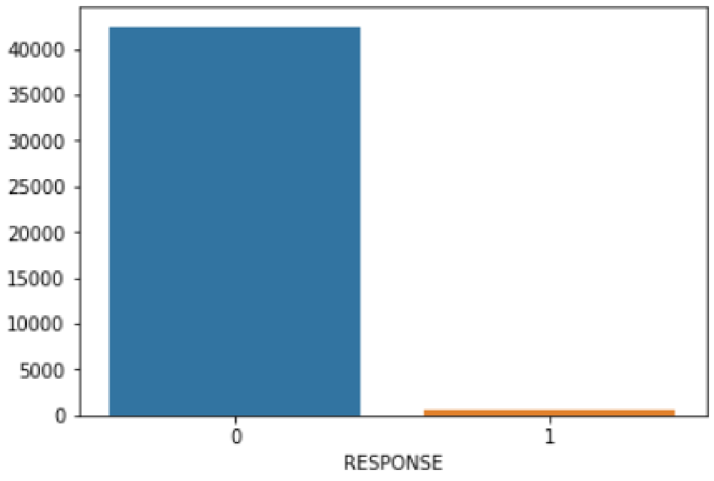
ONLINE_PURCHASE	0	1
Cluster		
0	1255	71
1	4924	777
2	495	300
3	958	299
4	59813	5559
5	5993	1646
6	2905	308
7	23273	3992
8	45026	3066
9	4155	250
10	25559	1028

Another interesting aspect is that in relation to the general population within each cluster, we can notice that cluster 6 has the highest percentage of customers. Therefore, cluster 6 could also become a potential target group.

3.3 Supervised learning

In the final task, the objective was to create a supervised model which could predict the likelihood of a customers responding to a mail-out campaign. Several approaches were taken and compared, including logistic regression, random forest and XGBoost.

Investigating the distribution of the target variable, one could note there were imbalance in the classes.



In order to resolve this issue, upsampling of the minority class was performed. However, for random forest and XGBoost, this could also be handled by specifying the following two parameters **class_weight** (random forest) and **scale_pos_weight** (XGBoost) which then takes into account the class imbalance.

As a final step, each model was trained on the provided training data (after applying all the data pre-processing steps described in the previous subsection) using the default parameters in the model classes. The ROC for each model is shown in the table below.

Algorithm	ROC
Logistic regression	0.56
XGBoost	0.55
Random forest	0.49

4 Conclusion

In conclusion, the final project contained two different parts, a customer segmentation part as well as a classification task to predict the customers that are most likely to respond to a mail-out campaign. General data pre-processing steps was performed and PCA was applied. Thereafter, a KMeans model was built in order to identify existing clusters in the general population of Germany. With the help of our customer dataset, one could then find similar individuals in the general population. The overall results indicated that cluster 4 and 8 were the ones with highest potential. However, additional business aspects has to be taken into consideration, for instance if there were a specific type of individual one would like to target, e.g. online buyers. In terms of the classification problem, several models were compared, including Logistic regression, Random forest and XGBoost. In order to tackle the imbalanced data, upsamplig was utilized. At the end, the best performing model using ROC as the target measure was Logistic regression.

5 Further improvements

First and foremost, just throwing 94 features away because they lack a description in a real life project would not be considered a "correct" way of handling it. Before doing that, one should at least talk to the owners of the data and try to get additional information, even though it was not delivered in the excel files. Another aspect that also could be improved is the handling of missing values, instead of just imputing -1, one could instead investigate each column and come up with an unique solution for each column, e.g. imputing with mean, median or even building a model to impute the missing value. Lastly, in order to improve the models for the classification task, a hyperparameter search would be useful.