Capstone Report to Customer Segmentation – Arvato Financial Solutions

Hung Thai Le

Abstract – In this proposal, I first present a solution to the customer segmentation task of a mail-order sales company, which helps identify the common characteristics of customers as well as of non-customers. Then, I leverage results obtained from the segmentation task to build a model that can predict which individuals among the company's targets will respond to the company. The prediction result on the test set has been submitted to Kaggle and I achieved a score of 0.79208 that is only 0.01855 smaller than the best score recorded at that time.

Table of Contents

I. Definition

1. Project Overview

Customer segmentation has become one of the first and foremost task that must be done by any large company before it starts marketing campaigns. This complex task can be implemented by the company itself or by a third-party. Arvato, as a global third-party, has been providing its clients innovative solutions with a focus on automation and data analytics

In the context of this Machine Learning Nanodegree Program, we work with Arvato's datasets to provide a solution to the customer segmentation/acquisition problem of a mail-order sales company that sells organic products in Germany. Specifically, Arvato provides four demographic files. Each file contains one dataset and in CSV (Comma-separated values) format, in which each row represents demographics of an individual and each row represents an attribute.

- Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891,211 persons (rows) x 366 features (columns),

- Udacity_CUSTOMERS_052018.csv: Demographics data for customers of the mail-order company; 191,652 persons (rows) x 369 features (columns),
- Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42,982 persons (rows) x 367 (columns),
- Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42,833 persons (rows) x 366 (columns).

Besides those data files, Arvato also provides us the two additional metadata files that contain attributes' information:

- DIAS Information Levels - Attributes 2017.xlsx: list of attributes and descriptions,
- DIAS Attributes - Values 2017.xlsx: detailed mapping of data values for each attribute.

For the ease of analysis, the two above files were converted into CSV format. The detailed data analysis is presented in Section II.

In this project, I use a machine learning algorithm to perform customer segmentation on the first two dataset. After that, I apply what I have found from the segmentation task on the third dataset and then develop a model to predict which individuals are most likely to convert into becoming customers for the company. The proposed model is applied on the fourth dataset, its outputs are submitted to Kaggle competition.

2. Problem Statement

Intuitively, the original problem is: Given the set of customers, how can the mail-order sales company acquire new customers? Fortunately, Arvato already did the very first step, i.e. to collect the demographics data for (i) customers and for (ii) the general population. Therefore, we can start to analyze the both demographics and perform customer segmentation to identify:

- Which parts of the population that best describe the core customer base of the company,
- Which demographic features are most common among customers?

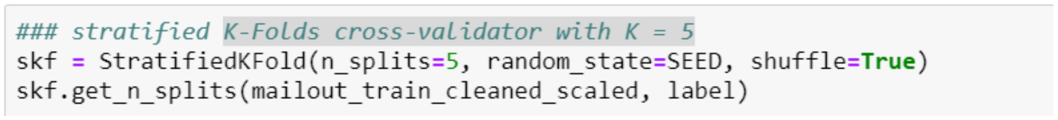
In this segmentation task, unsupervised machine learning technique is used to perform clustering.

After customer segmentation, we can apply the findings on the third dataset in the customer acquisition task. I develop a supervised model to predict which individuals are most likely to convert into becoming customers for the company. Multiple proposed models are then evaluated but only the model with the highest score is selected and submitted to Kaggle competition.

3. Evaluation Metrics

For the two stepped problem (i.e., customer segmentation and customer acquisition), I propose a two stepped solution. In the customer segmentation task, it is difficult to define an evaluation metric because the outputs are insights from datasets (e.g., the common characteristics of individuals who potentially become new customers) that help the Arvato's client make decisions in its campaign strategy.

As for the customer acquisition part, the straight forward way is to use accuracy as an evaluation metric. However, I found that there is a large output class imbalance where almost all individuals did not respond to the mailout. Specifically, there are 42,430 observations with label '0' while only 532 observations with label '1'. Therefore, instead of accuracy, I use AUROC (Area under the Receiver Operating Curve) as an evaluation metric. Also, the cross validation is applied to assess how the results of a model will generalize to an independent data set. I use K-Folds cross-validator with K = 5.



```
### stratified K-Folds cross-validator with K = 5
skf = StratifiedKFold(n_splits=5, random_state=SEED, shuffle=True)
skf.get_n_splits(mailout_train_cleaned_scaled, label)
```

II. Analysis

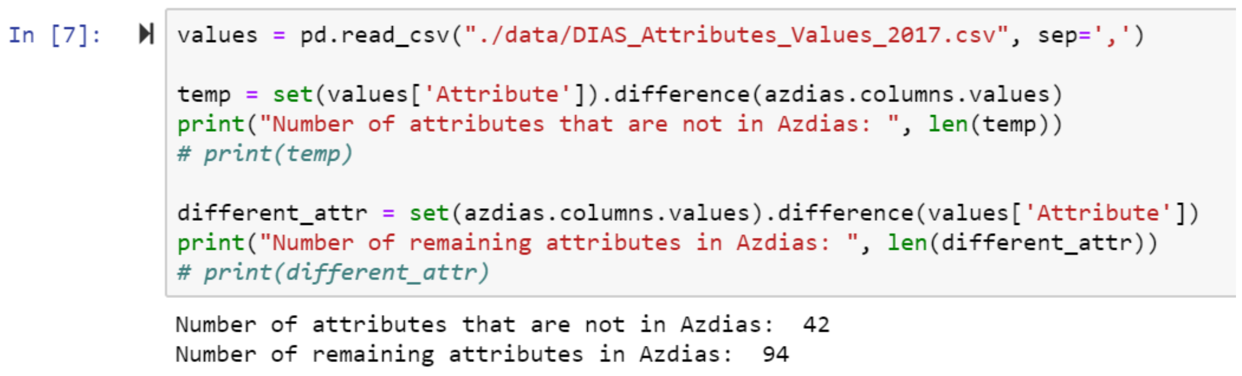
My analysis session starts with data exploration, where initial findings from the dataset are presented. Note that I will address those issue later in data preprocessing part of Section III. In the second part of the analysis session, I briefly present a discussion of main techniques used in the project. Finally, the benchmark model is given.

In the associated Jupyter Notebook, I assume that all the four datasets are in a folder named "***data***". Hereinafter, the first and second datasets, which are used in the customer segmentation task, are referred to as ***Azdias*** and ***Customers*** dataset, respectively. The two additional metadata files that contain Azdias attributes' information are referred to as the ***Azdias metadata*** files. Also, it should be noted that I flexibly use two terms "attribute" and "feature" based on the context.

1. Data Exploration and visualization

1.1. Attribute name

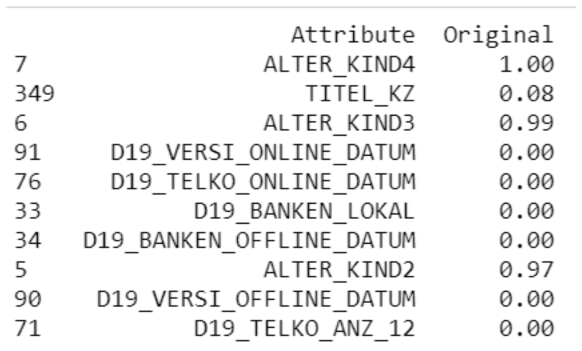
It is seen that the number of attributes in the Azdias dataset is different from the number of attributes described in the Azdias metadata. In fact, there are 94 attributes existing in the Azdias dataset have no detailed information.



```
In [7]:   values = pd.read_csv("./data/DIAS_Attributes_Values_2017.csv", sep=',')

          temp = set(values['Attribute']).difference(azdias.columns.values)
          print("Number of attributes that are not in Azdias: ", len(temp))
          # print(temp)

          different_attr = set(azdias.columns.values).difference(values['Attribute'])
          print("Number of remaining attributes in Azdias: ", len(different_attr))
          # print(different_attr)

          Number of attributes that are not in Azdias:  42
          Number of remaining attributes in Azdias:  94
```

1.2. Unknown values

Ideally, the dataset should not contain unknown values because those values damage the sclustering process. However, it is difficult to collect such datasets since it may contain sensitive information. Looking at the Azdias dataset, I found that there is a large number of unknown values:



	Attribute	Original
7	ALTER_KIND4	1.00
349	TITEL_KZ	0.08
6	ALTER_KIND3	0.99
91	D19_VERSI_ONLINE_DATUM	0.00
76	D19_TELKO_ONLINE_DATUM	0.00
33	D19_BANKEN_LOKAL	0.00
34	D19_BANKEN_OFFLINE_DATUM	0.00
5	ALTER_KIND2	0.97
90	D19_VERSI_OFFLINE_DATUM	0.00
71	D19_TELKO_ANZ_12	0.00

The above table shows the proportion of unknown values of some attributes (in the column named 'Original') before applying any feature engineering. We can see that the propositions given the attribute of ALTER_KIND4, ALTER_KIND3 are very high.

1.3. Data type

Most attributes are in ordinal data type naturally. The other data types in the Azdias dataset include categorical, binary and numerical data. Unfortunately, some attributes have no description so we cannot categorize them.

1.4. Others

From the set of available values of all attributes, I found some interesting points that must be addressed in the data preprocessing.

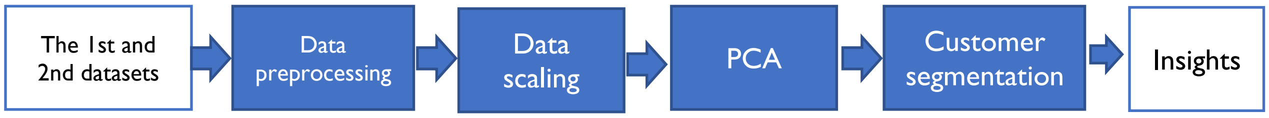
- LNR is unique for each row and it is not feature
- CAMEO_DEUG_2015 data is mixed between string and float
- CAMEO_DEU_2015 data has the set of string values (1A, 1B, …, 9E).
- OST_WEST_KZ has binary values but they are in different formats of ['O', 'W'], not [0, 1].

- EINGEFUEGT_AM is in date-time format, making difficult to process data in clustering
- LP_FAMILIE_FEIN, LP_FAMILIE_GROB, LP_LEBENSPHASE_FEIN, LP_LEBENSPHASE_GROB, ORTSGR_KLS9, KBA05_MODTEMP have values that were not described in the Azdias metadata.

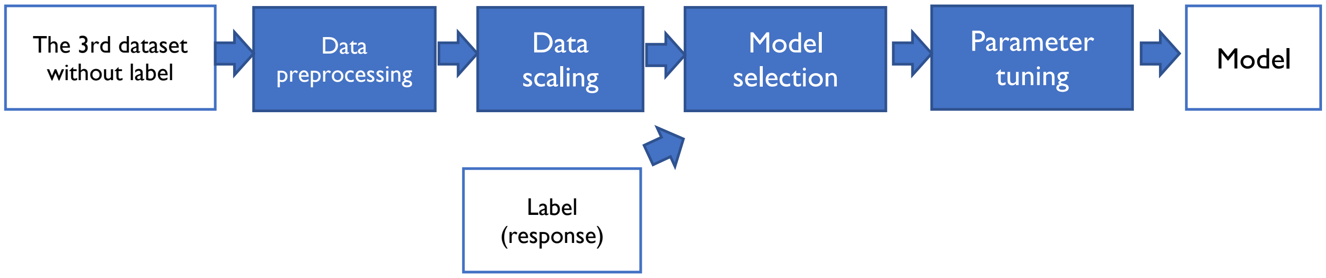
2. Algorithms and Techniques

2.1. Process workflow

In the customer segmentation task, the two datasets are cleaned at the beginning in data preprocessing. After scaling data (optional) and reducing the number of dimensions by PCA (Principal Component Analysis), I perform customer segmentation using the K-Means algorithm. Based on obtained results, I can find (i) the common characteristics among customers in a group and (ii) which parts of the population are most likely to convert into becoming customers in the future.



In the customer acquisition task, the output of the data scaling process is the input of a model selection. I first examine multiple algorithms with default parameters and then select the one that has the best score. In the parameter tuning step, the random search algorithm is used to decide the selected model's hyperparameters. Finally, the best model is used to make decisions on the test dataset and the results will be submitted to Kaggle.



In the following parts of Subsection III.3, I discuss the techniques used in the project in more details, including data scaling, dimensionality reduction, clustering algorithms, and supervised algorithms.

2.2. Data Scaling

When working with a learning model, it is important to scale the features to a range which is centered around zero. One reason is that if a feature's variance is larger than the variance of other features, that particular feature might dominate other features that we do not want it happens in our model. On the other words, we need scale all data so that the variance of the features is in the same range

Standard scaler and Minmax scaler are two popular scalers for this purpose, In the project, I adopt the standard scaler, the implementation with Minmax scaler will be lefts for future work.

2.3. Dimensionality Reduction

When the number of features increases, we often face overfitting issues (a model becomes too heavy to generalize data). Dimensionality reduction is a very important technique that helps us to create models that are lighter but more accurate thanks to the reduction of misleading data. In addition, it helps to reduce computing power and storage so the training time is shortened.

The popular methods used for dimensionality reduction include:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Generalized Discriminant Analysis (GDA)

In this project, I use the PCA algorithm and leave the other for future work.

2.4. Clustering

To date, there are various algorithms that can be applied to divide the general population and customer population into different clusters. A list of 10 of the popular algorithms is as follows:

- Affinity Propagation
- Agglomerative Clustering
- BIRCH
- DBSCAN
- K-Means
- Mini-Batch K-Means
- Mean Shift
- OPTICS
- Spectral Clustering
- Mixture of Gaussians

In this project, K-Means clustering is selected since this algorithm is simple and the implementation is not complex. About K-Means clustering, the main idea of this algorithm is to divide all observations into K clusters such that the sum of the squared distance between observations and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum.

2.5. Supervised algorithms

In the customer acquisition task, because we are aware of the response of customers in the past, it is straightforward to use a supervision algorithm rather than unsupervised algorithms. I examine the following algorithms:

- Logistic Regression
- Random Forest Classifier
- XGB Classifier
- Light Gradient Boosted Machine
- Gradient Boosting Classifier
- Multi-layer Perceptron classifier

As the explanations of the above supervised algorithms require advanced machine learning knowledge, I will leave this work in the future. In implementations for my evaluations, I use the default settings for each algorithm (but the same setting of random_state). The algorithm that has the best score will be use in the refinement part.

3. Benchmark model

In this proposal, I select the Logistic Regression model as a baseline model for the customer acquisition task since it is simple and easy to train within less time. To date, 0.81063 is the best score recorded by Kaggle.

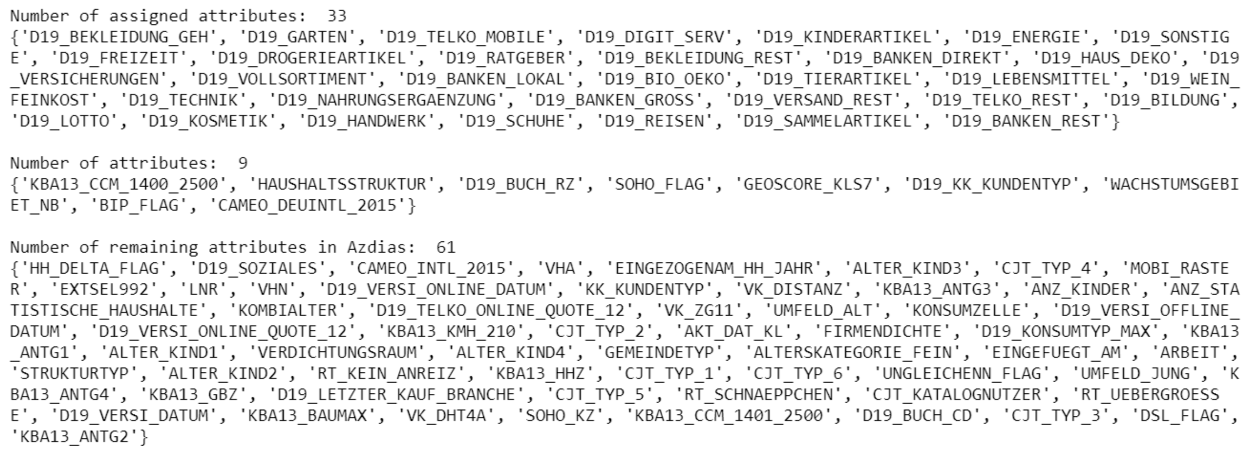
III. Methodology

In this third session, I present my implementation for each task in detail, starting with data preprocessing. Finally, the obtained results and discussion are also given.

1. Data Preprocessing

1.1. Attribute name correction

As mentioned earlier in the data exploration part, there are 94 attributes in the Azdias dataset has no detailed information. This issue may happen in real projects and should be corrected by the team. As I have no change to ask for corrections, I took a look in the Azdias dataset and found that some attributes names are missed “_NZ”. Overall, 33 attributes’ names are re-assigned


```
Number of assigned attributes:  33
{'D19_BEKLEIDUNG_GEH', 'D19_GARTEN', 'D19_TELKO_MOBILE', 'D19_DIGIT_SERV', 'D19_KINDERARTIKEL', 'D19_ENERGIE', 'D19_SONSTIG
E', 'D19_FREIZEIT', 'D19_DROGERIEARTIKEL', 'D19_RATGEBER', 'D19_BEKLEIDUNG_REST', 'D19_BANKEN_DIREKT', 'D19_HAUS_DEKO', 'D19
_VERSICHERUNGEN', 'D19_VOLLSORTIMENT', 'D19_BANKEN_LOKAL', 'D19_BIO_OEKO', 'D19_TIERARTIKEL', 'D19_LEBENSMITTEL', 'D19_WEIN_
FEINKOST', 'D19_TECHNIK', 'D19_NAHRUNGSERGAENZUNG', 'D19_BANKEN_GROSS', 'D19_VERSAND_REST', 'D19_TELKO_REST', 'D19_BILDUNG',
'D19_LOTTO', 'D19_KOSMETIK', 'D19_HANDWERK', 'D19_SCHUHE', 'D19_REISEN', 'D19_SAMMELARTIKEL', 'D19_BANKEN_REST'}

Number of attributes:  9
{'KBA13_CCM_1400_2500', 'HAUSHALTSSTRUKTUR', 'D19_BUCH_RZ', 'SOHO_FLAG', 'GEOSCORE_KLS7', 'D19_KK_KUNDENTYP', 'WACHSTUMSGEBI
ET_NB', 'BIP_FLAG', 'CAMEO_DEUINTL_2015'}

Number of remaining attributes in Azdias:  61
{'HH_DELTA_FLAG', 'D19_SOZIALES', 'CAMEO_INTL_2015', 'VHA', 'EINGEZOGENAM_HH_JAHR', 'ALTER_KIND3', 'CJT_TYP_4', 'MOBI_RASTE
R', 'EXTSEL992', 'LNR', 'VHN', 'D19_VERSI_ONLINE_DATUM', 'KK_KUNDENTYP', 'VK_DISTANZ', 'KBA13_ANTG3', 'ANZ_KINDER', 'ANZ_STA
TISTISCHE_HAUSHALTE', 'KOMBIALTER', 'D19_TELKO_ONLINE_QUOTE_12', 'VK_ZG11', 'UMFELD_ALT', 'KONSUMZELLE', 'D19_VERSI_OFFLINE_
DATUM', 'D19_VERSI_ONLINE_QUOTE_12', 'KBA13_KMH_210', 'CJT_TYP_2', 'AKT_DAT_KL', 'FIRMENDICHTE', 'D19_KONSUMTYP_MAX', 'KBA13
_ANTG1', 'ALTER_KIND1', 'VERDICHTUNGSRAUM', 'ALTER_KIND4', 'GEMEINDETYP', 'ALTERSKATEGORIE_FEIN', 'EINGEFUEGT_AM', 'ARBEIT',
'STRUKTURTYP', 'ALTER_KIND2', 'RT_KEIN_ANREIZ', 'KBA13_HHZ', 'CJT_TYP_1', 'CJT_TYP_6', 'UNGLEICHENN_FLAG', 'UMFELD_JUNG', 'K
BA13_ANTG4', 'KBA13_GBZ', 'D19_LETZTER_KAUF_BRANCHE', 'CJT_TYP_5', 'RT_SCHNAEPPCHEN', 'CJT_KATALOGNUTZER', 'RT_UEBERGROESS
E', 'D19_VERSI_DATUM', 'KBA13_BAUMAX', 'VK_DHT4A', 'SOHO_KZ', 'KBA13_CCM_1401_2500', 'D19_BUCH_CD', 'CJT_TYP_3', 'DSL_FLAG',
'KBA13_ANTG2'}
```

From the result, there are 61 remaining attributes that are not clear. They are named in Germany and are encoded in short forms. Since it is difficult to understand their meanings, all unclear attributes are kept for future processing.

1.2. Unknown value handling

In the project, I remove rows/columns that contains a lot of unknown values and do some engineering in rows/columns that have small number of unknow values. The detailed workflow of this process is as follows.

- Manually create a list of unknown values of attributes (from the Azdias metadata and the previous data exploration)
- Drop the columns/rows that contains a large number of unknown values
- Convert unknown values into other values (to prepare for segmentation). In my case, I use the most frequent value per each attribute.

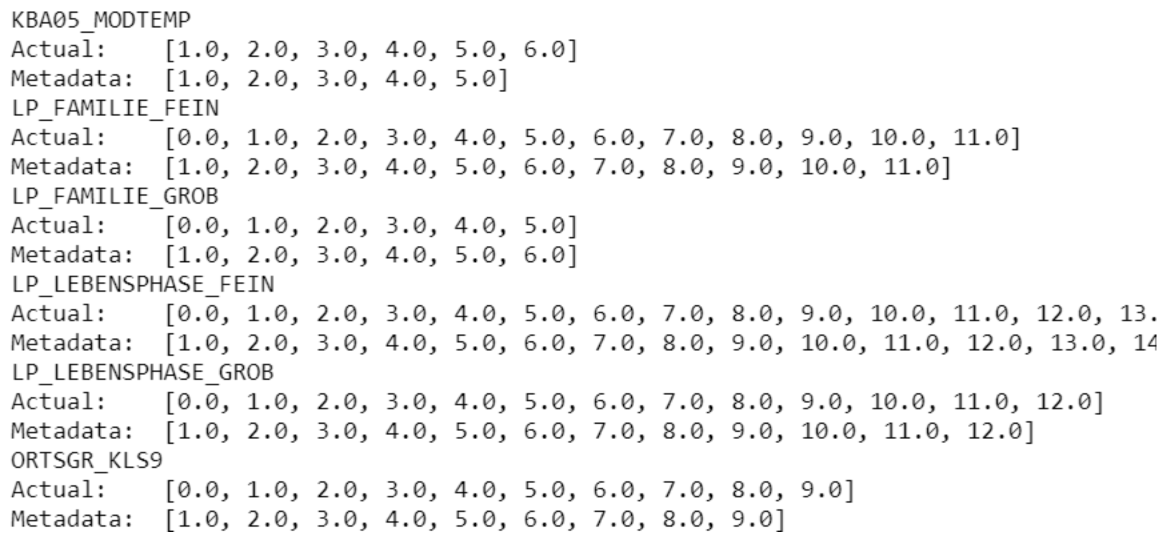
1.2.1. Create a list of unknown values of attributes

In the Azdias dataset, the unknown values are stored in different formats. Most information can be found from the Azdias metadata. Besides, I guess the corresponding unknown values by considering:

a) similar attributes (e.g., unknown values of attribute 'KBA13_ANTG1' is highly similar to those of attribute 'KBA05_ANTG1'),

b) set of possible values: (e.g. ‘X’ will be an unknown value if other values are integers of -1, 0, 1, …, 9).

It should be noted that there are attributes whose values exist in the Azdias dataset but were not in its metadata. The following list shows attributes whose actual values are different from the given values in metadata file:



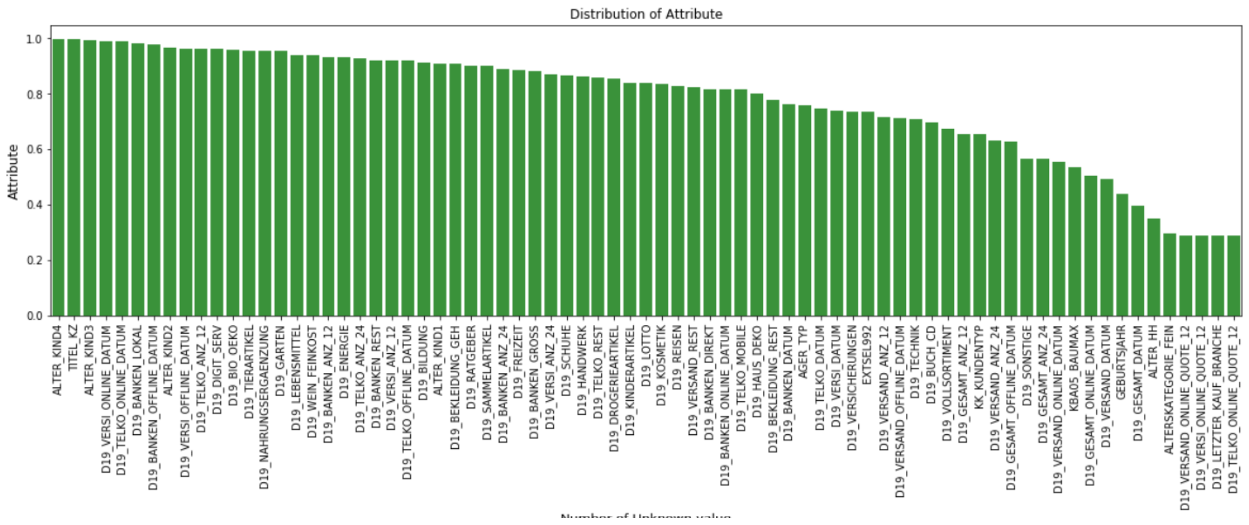
```
KBA05_MODTEMP
Actual:    [1.0, 2.0, 3.0, 4.0, 5.0, 6.0]
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0]
LP_FAMILIE_FEIN
Actual:    [0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0]
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0]
LP_FAMILIE_GROB
Actual:    [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0, 6.0]
LP_LEBENSPHASE_FEIN
Actual:    [0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14
LP_LEBENSPHASE_GROB
Actual:    [0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0]
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0]
ORTSGR_KLS9
Actual:    [0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0]
Metadata:  [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0]
```

Because it is difficult to decide available values of KBA05_MODTEMP, I remove the corresponding column from the dataset. As for other attributes (e.g., LP_FAMILIE_FEIN), I consider 0.0 as an unknown value.

The final list of attributes and their unknown values are stored in a file named ***feature-final.csv***. It should be noted that manual process without any confirmation is not recommended in production. In real projects, the team should rise a ticket to find the exact unknown values.

1.2.2. Drop columns that contains a large number of unknown values

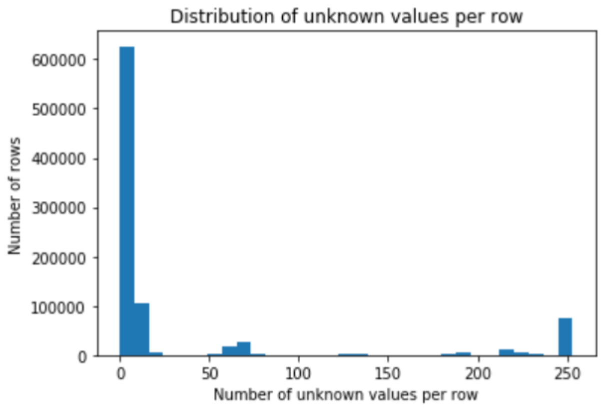
We first convert all unknown values into the same value (i.e. np.NaN in my case). The following figure shows 75 columns that contain the highest number of unknown values.



In this project, I decided to drop all columns whose unknown values account for more than 30% of total values. Overall, there are 70 columns removed from the dataset.

1.2.3. Drop rows that contains a large number of unknown values

The following figure shows the distribution of unknown values per row. From that figure, I decide to drop all rows of which the number of unknown values account for more than 30% of total values (i.e. 89). Overall, 88% of rows are removed from the Azdias dataset.



1.2.4. Impute unknown values

After removing the features and rows which contained unknown values, we will replace all unknown values with another value. Since the Azdias dataset corresponds to the population in general, imputing the unknown values with the most frequent observations is reasonable. Furthermore, this way is simple and straightforward.

1.3. Feature engineering

In this part, we make decisions on what attributes are used in the future tasks (i.e., customer segmentation and customer acquisition). I firstly drop attributes because they are unclear or highly correlated to other attributes:

- KBA05_MODTEMP: As mentioned earlier, this attribute contains an unclear value of 6
- GEMEINDETYP, D19_KONSUMTYP_MAX, KOMBIALTER, VERDICHTUNGSRAUM has no information and their set of values are unclear
- LNR is unique for each row and it is not feature
- CAMEO_DEU_2015 and CAMEO_DEUG_2015 are both describing New German CAMEO Typology but at different scales. Since CAMEO_DEU_2015 contains 44 categories, I decide to keep CAMEO_DEUG_2015 only and drop CAMEO_DEU_2015.
- Similar to CAMEO_DEUG_2015, I decide to keep LP_LEBENSPHASE_GROB and remove LP_LEBENSPHASE_FEIN.

Besides, the following attributes contain mixed data, I decide to separate each of them into multiple attributes:

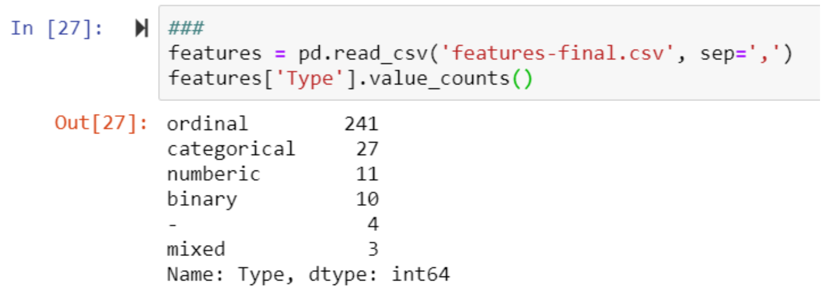
- CAMEO_INTL_2015 is separated into WEALTH and LIFE_AGE
- PRAEGENDE_JUGENDJAHRE is separated into GENERATION_DECADE and MOVEMENT

1.4. Data type

Given the selected attributes after attribute engineering, we do additional operation to ensure all values of all attributes are in formats that make further processing easier.

- CAMEO_DEUG_2015 is converted from string to float
- OST_WEST_KZ has binary values but they are in different representations. Those values are converted from ‘O’ and ‘W’ to 0 and 1, respectively
- EINGEFUEGT_AM is in date-time format. Since it is continuous data, I convert date-time data to date format only

After the conversions, all attributes are manually divided into four groups: ordinal, categorical, binary and numerical data group. The detailed information can be found in feature-final.csv. Note that in that CSV file, the attributes categorized as 'mixed' and '-' (i.e. dropped) can be ignorable.



```
In [27]:  ###
          features = pd.read_csv('features-final.csv', sep=',')
          features['Type'].value_counts()

Out[27]:  ordinal        241
          categorical     27
          numberic        11
          binary          10
          -                4
          mixed            3
          Name: Type, dtype: int64
```

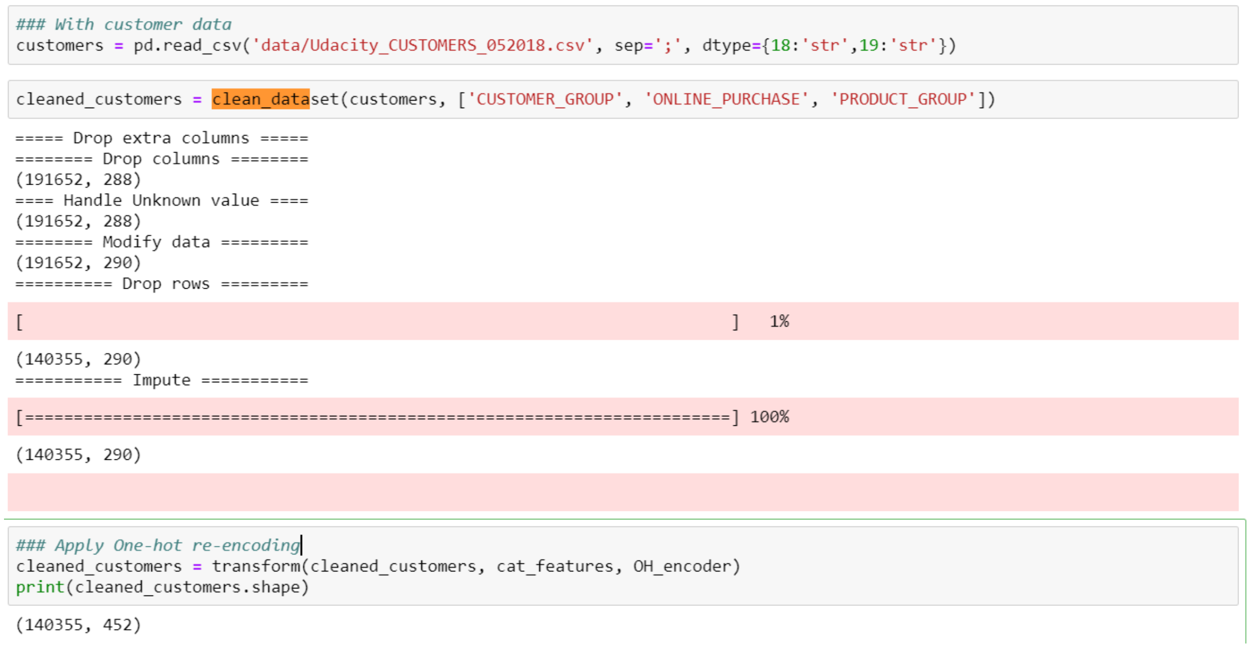
Among the four groups, the categorical group can be re-encoded using the one-hot encoding. The shape of the Azdias dataset after the above processes is (785302, 452)

1.5. Summary of data preprocessing

In summary of data pre-processing, the given dataset is processed following the workflows:

1. Drop columns/attributes including new attributes (i.e. attributes do not exist in Azdias), unclear attributes, and attributes that contain a lot of np.NaN.
2. Get unknown values of each attribute from ***feature-final.csv***.
3. Convert unknown values into np.NaN.
4. Do feature engineering
5. Drop rows that contains a lot of np.NaN
6. Convert np.NaN (representing unknown values) into the most frequent value by column

The obtained output of the customer dataset is shown in the following figure.

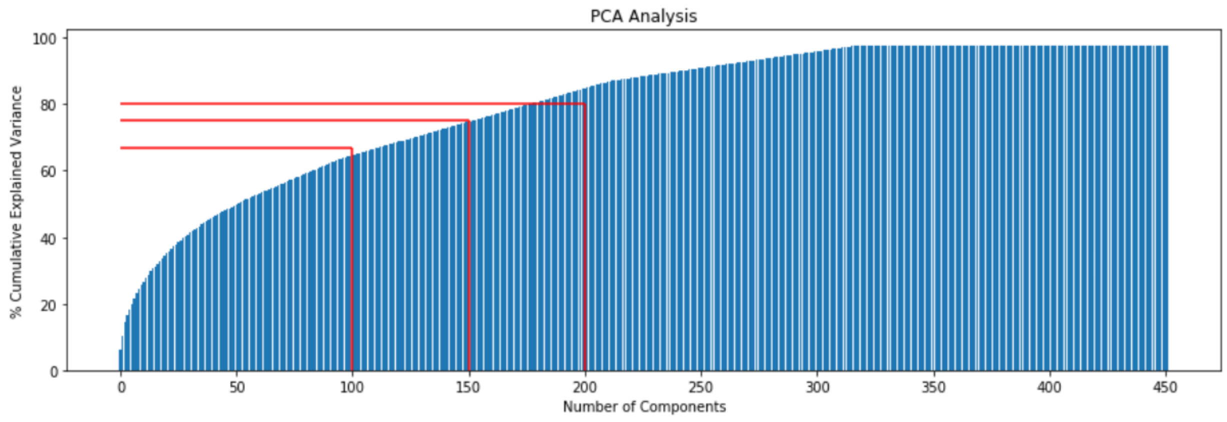


```
### With customer data
customers = pd.read_csv('data/Udacity_CUSTOMERS_052018.csv', sep=';', dtype={18:'str',19:'str'})

cleaned_customers = clean_dataset(customers, ['CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP'])

===== Drop extra columns =====
======== Drop columns ========
(191652, 288)
==== Handle Unknown value ====
(191652, 288)
======== Modify data =========
(191652, 290)
========== Drop rows =========
[                                                                        ]   1%
(140355, 290)
=========== Impute ===========
[========================================================================] 100%
(140355, 290)

### Apply One-hot re-encoding
cleaned_customers = transform(cleaned_customers, cat_features, OH_encoder)
print(cleaned_customers.shape)

(140355, 452)
```

2. Implementation for customer segmentation

2.2. PCA

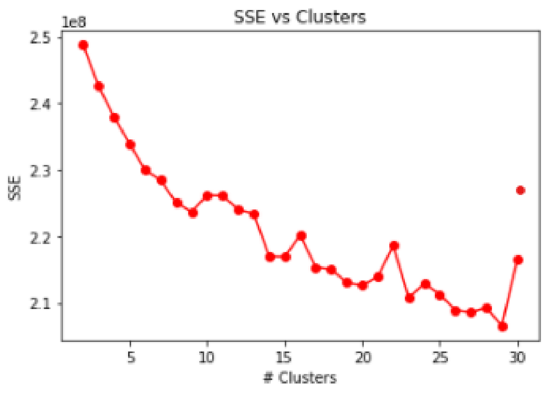
After applying standard scaling, we start implementing PCA. The main question is how many components should be reduced to from the original number of 452. The following figure shows the percent of explained variance given a number of components. The red lines are associated with a cumulative explained variance of 0.5, 0.75, and 0.8.



It is seen that the percent increases fast at the beginning but remains unchanged when the number of components is higher than 300. I decide to use 150 components because the related percent of explained variance is about 0.75 (i.e., 75%) and the number of components is not too large.

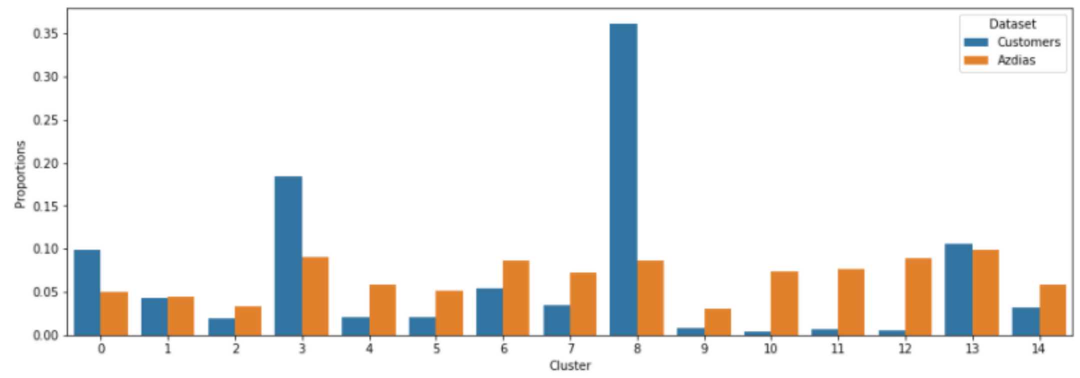
2.3. K-Means

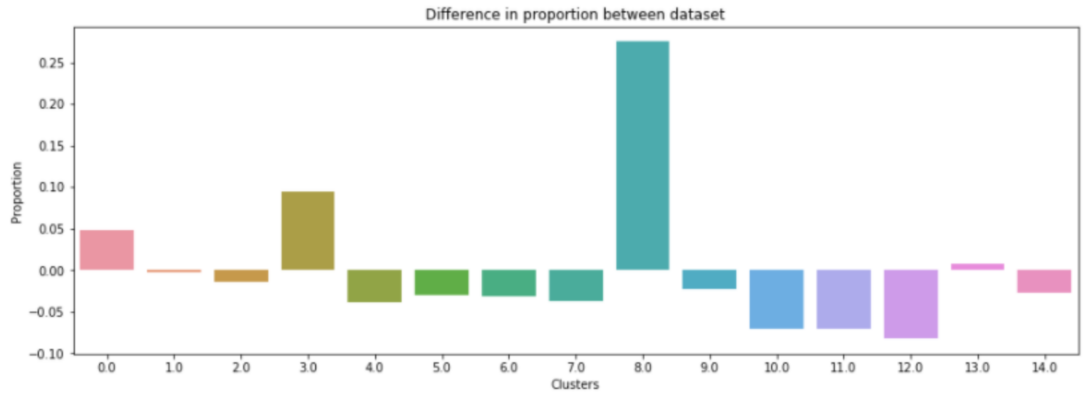
In the K-Means algorithm, the number of clusters is the most important hyperparameter. We select the number of clusters in order to minimize the intra-cluster variation. Unfortunately, there is no definitive way of selecting the number of clusters. In this project, I apply Elbow method in which SSE (sum of squared distances in each cluster for the specified list of number of clusters) is plotted to help decide the number of clusters



The above figure helps us understand how the number of clusters affect the intra-cluster distances. From the figure, I roughly select a number of 15.

After applying the K-Means algorithm again with a number of clusters being 15, I get the results of the clustering task on the two datasets (i.e., customers and Azdias). In the following figures, the former figure shows the proportion of individuals for each cluster while the latter figure shows the proportion difference (the positive difference means the proportion in the customer dataset is higher than that of the Azdias dataset).





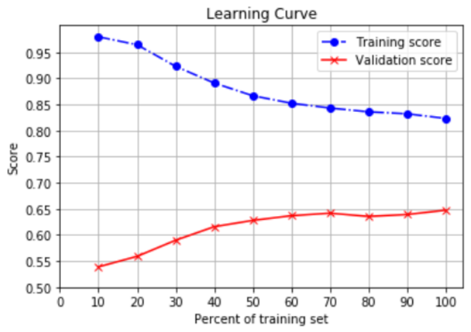
Looking at the results, I found that cluster 8 and 12 has the highest/lowest difference proportions. In the other words, the company should focus on individuals in cluster 8 because they likely become new customers in the future. Hereinafter, I refer to cluster 8 and cluster 12 as the target and general group, respectively.

3. Implementation and Refinement for customer acquisition

3.1. Performance evaluations

In the customer acquisition task, I also apply Standard scaler. After that, I implement the 6 methods with default values. The following figures show the results given by each algorithm.

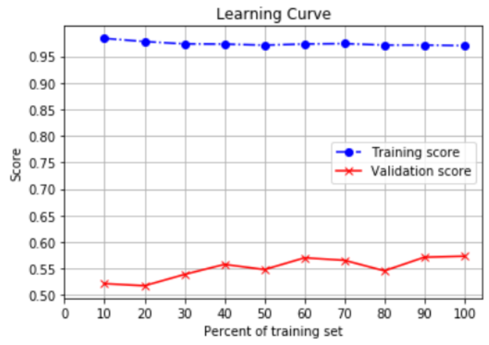
```
LR: 0.647448 (std: 0.012088)
ROC-AUC train score = 0.82
ROC-AUC validation score = 0.65
```



```
RF: 0.570925 (std: 0.019804)
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.6/s
g: A worker stopped while some jobs were given to the
ory leak.
  "timeout or by a memory leak.", UserWarning
```

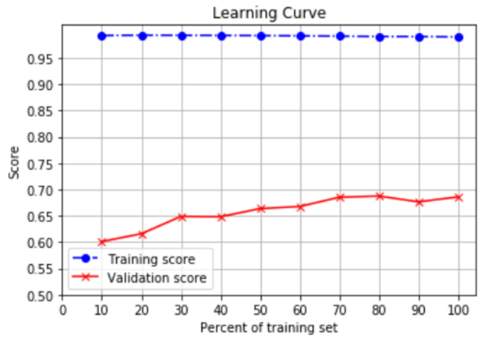
```
ROC-AUC train score = 0.97
ROC-AUC validation score = 0.57
```



```
XGB: 0.681228 (std: 0.016856)
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.6/s
g: A worker stopped while some jobs were given to the
ory leak.
  "timeout or by a memory leak.", UserWarning
```

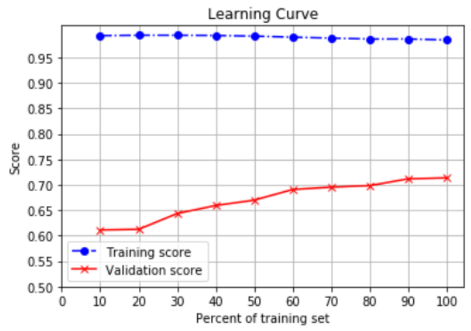
```
ROC-AUC train score = 0.99
ROC-AUC validation score = 0.69
```



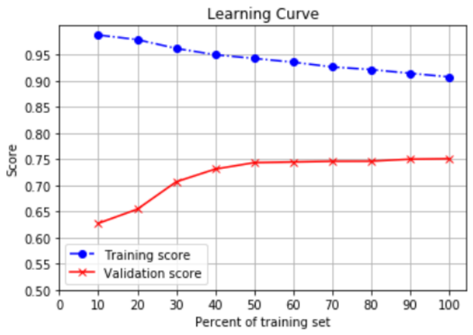
```
LGBM: 0.717584 (std: 0.018059)
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.6/si
g: A worker stopped while some jobs were given to the
ory leak.
  "timeout or by a memory leak.", UserWarning
```

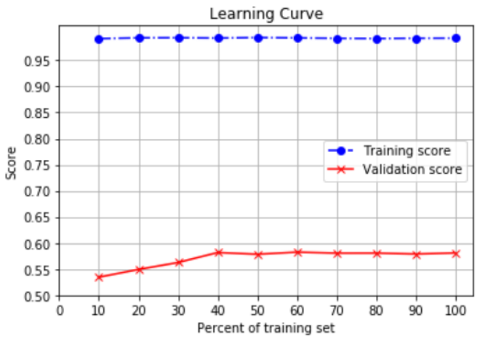
```
ROC-AUC train score = 0.98
ROC-AUC validation score = 0.71
```



```
GB: 0.749560 (std: 0.011148)
ROC-AUC train score = 0.91
ROC-AUC validation score = 0.75
```



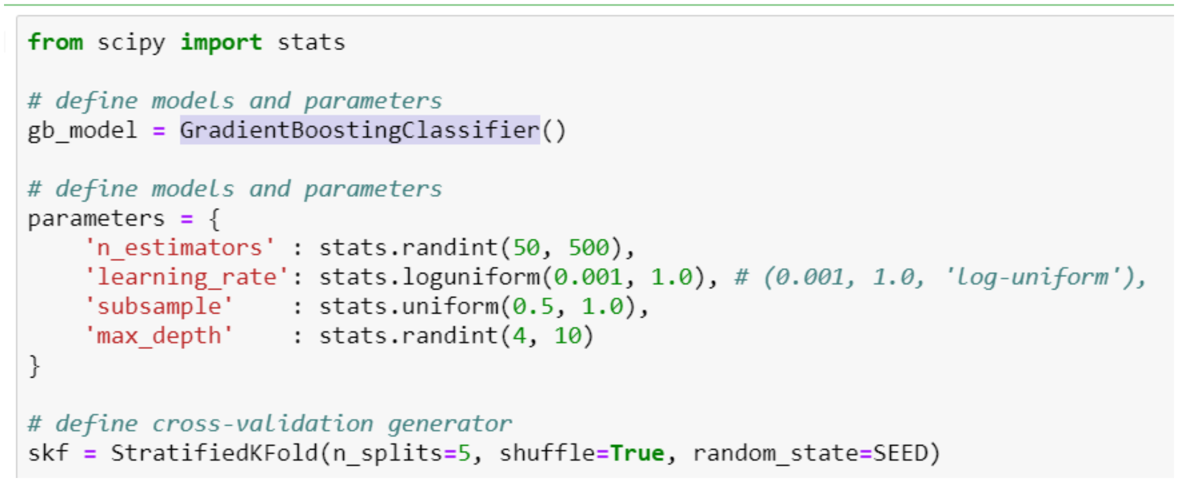
```
MLP: 0.581676 (std: 0.022023)
ROC-AUC train score = 0.99
ROC-AUC validation score = 0.58
```



The score of Logistic Regression (i.e., out selected baseline model) is 0.65. From the results of my evaluations, I decide to use GB (Gradient Boosting Classifier) since its score is the highest (0.75). The next part presents the hyperparameter tuning process.

3.2. Refinement for Gradient Boosting Classifier

For parameter tuning, I use random search to find the optimal set of hyperparameters. Compared to the grid search, this search method is comparatively less costly and more effective. As fully understanding all hyperparameters are out of the scope of this project, I only select the main four parameters to do tuning: n_estimators, learning_rate, subsample, and max_depth.



```python
from scipy import stats

# define models and parameters
gb_model = GradientBoostingClassifier()

# define models and parameters
parameters = {
    'n_estimators' : stats.randint(50, 500),
    'learning_rate': stats.loguniform(0.001, 1.0), # (0.001, 1.0, 'log-uniform'),
    'subsample'    : stats.uniform(0.5, 1.0),
    'max_depth'    : stats.randint(4, 10)
}

# define cross-validation generator
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=SEED)
```

The searching process with an iteration of 20 takes 1 hour 25 minutes. The obtained score is about 0.76 (i.e. about 1% higher than previous one).

IV. Results and Discussion

1. Customer Segmentation

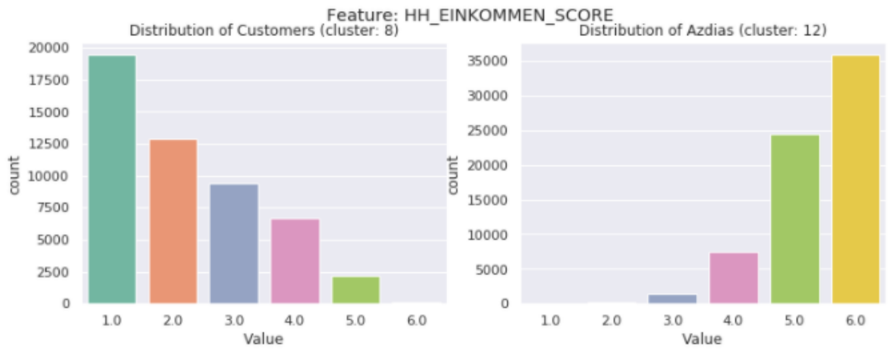
In this part, I analyze the target group (cluster 8) in the customer dataset to find out the common characteristics of current customers. For comparison purposes, the general group (cluster 12) in the Azdias dataset is also investigated.

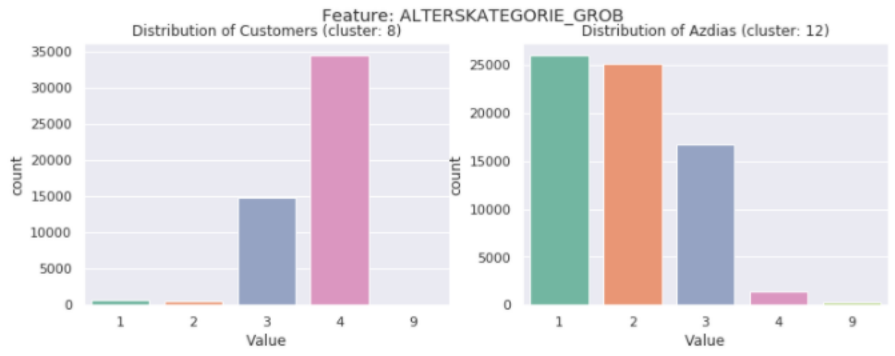
As Analyzing all attributes is an exhausted task, for learning purposes, I pick some attributes of which the distributions given the two datasets are clearly distinguished

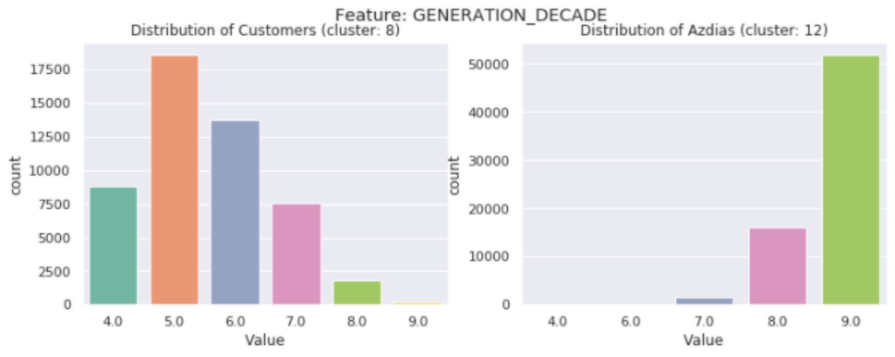
- HH_EINKOMMEN_SCORE: estimated household net income
- ALTERSKATEGORIE_GROB: age classification through prename analysis
- GENERATION_DECADE: separated from PRAEGENDE_JUGENDJAHRE
- FINANZ_SPARER: financial typology - money saver

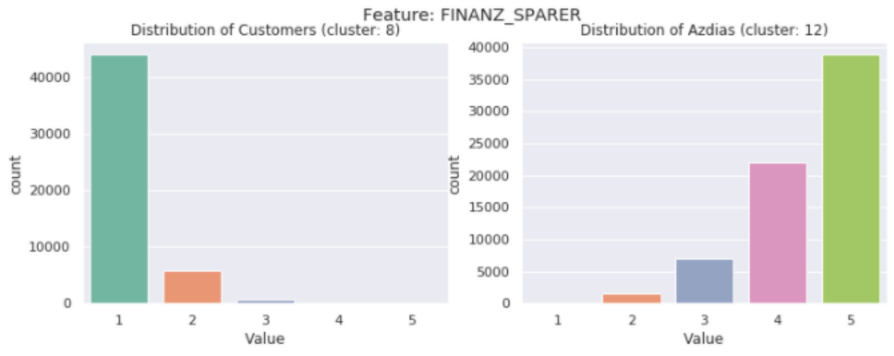
- MOBI_REGIO: moving pattern
- LP_STATUS_GROB: social status

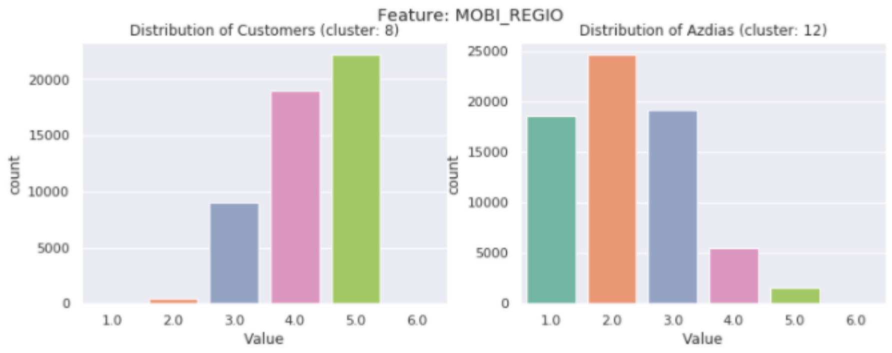
The results are shown in the following figures.

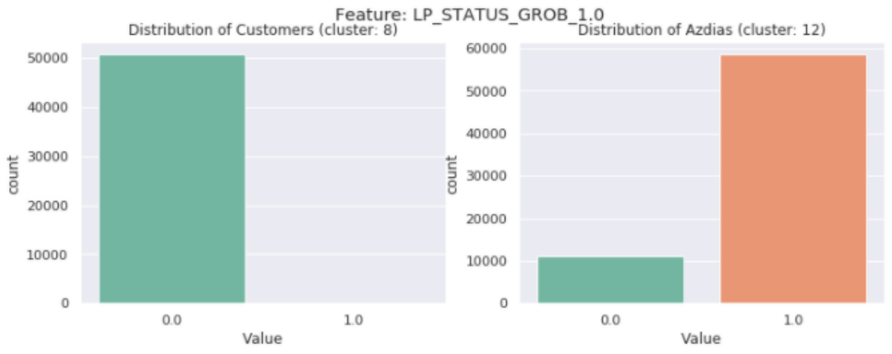


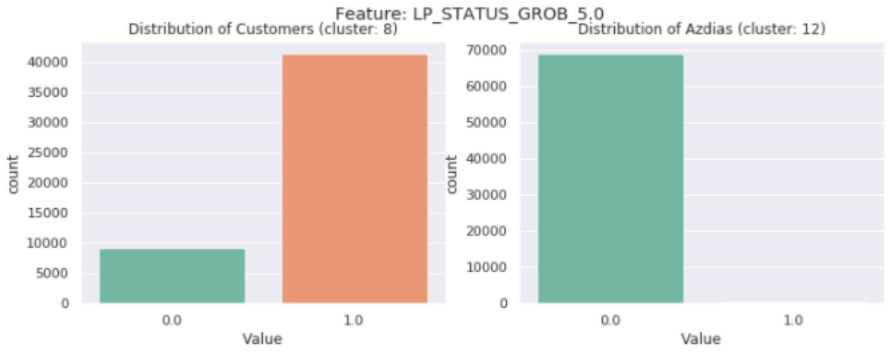








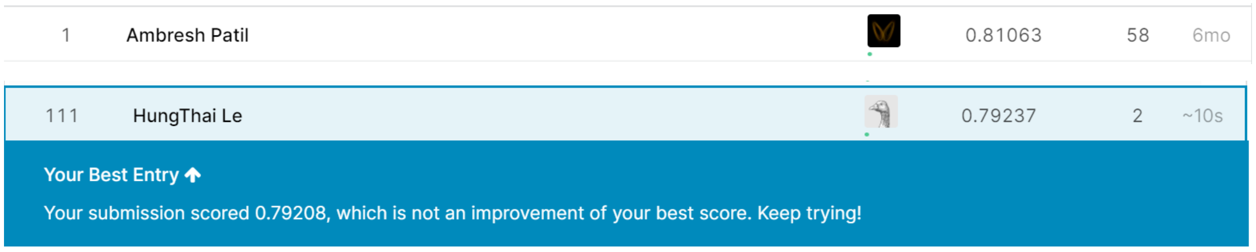




It is seen that the target customers likely have high net income (HH_EINKOMMEN_SCORE), lower moving patterns (MOBI_REGIO). They are 46 years old or even higher (ALTERSKATEGORIE_GROB, GENERATION_DECADE), potentially money savers (FINANZ_SPARER).

2. Customer Acquisition

The predictions on the test set are submitted to Kaggle and the resulting position and score on the test data is shown in the following figure.



1	Ambresh Patil	0.81063	58	6mo
111	HungThai Le	0.79237	2	~10s

Your Best Entry

Your submission scored 0.79208, which is not an improvement of your best score. Keep trying!

Compared to the highest score of 0.81063, my result is 0.79208. Although the result is about 0.01855 smaller, this is higher than the score on the training set (0.76). Furthermore, this is just my second submission. For learning purpose, the result is acceptable since my training time and computing power are limited. Interestingly, the first submission with XGB Classifier and without data scaling has a higher score (0.79237). So, the approach has some rooms to improve:

1. Select features used in the project
2. Collect feature information and divide them in precise types
3. Decide thresholds with some investigation rather a rough selection
4. Investigate techniques deeply to select more suitable algorithm, parameters

V. Acknowledgements

My sincere gratitude to Arvato Financial Solutions for the datasets/files and to Udacity for making this project possible.

I also would like to be thankful to Elena Ivanova for her post on https://towardsdatascience.com. Her post has motivated me to register the course as well as give me thorough ideas when I am doing the final project.