Capstone Project Report

Udacity Datascience Nanodegree
Customer Segmentation Report for Arvato Financial Services.

Author: Stephan Bauer

Date of Report: August, 2021

Project Definition

Overview

Arvato Financial Services is a company operating in the mail-order sales business in Germany. The company wants better identify persons that become more likely customers with their marketing campaigns.

Problem Statement

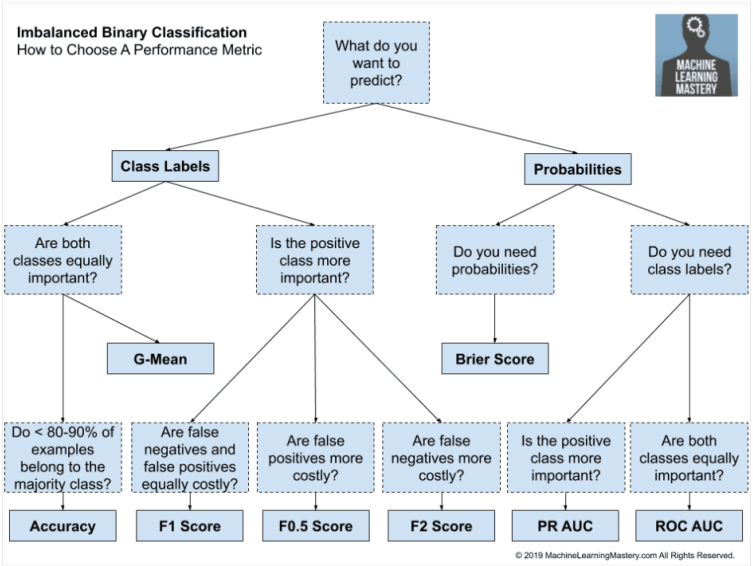
The Objective is to predict individuals who are more likely to become new customers from a given set of individuals.

To achieve this we use unsupervised and supervised machine learning approaches

- Unsupervised learning to segment non customers and customers into clusters to identify the main customer and population cluster and extract information about the driving features.
- Supervised learning to build and train a model to predict the probability of an individual to become a customer. For the supervised learning we will use different Binary Classifiers

Metrics

In order to find the correct metric this guide https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/ helps to find one:



- Data is very imbalanced
- We need probability

- We need class labels
- Both classes are important
- ⇨ This leads to **ROC AUC or PR-AUC**

The **Metric** used to evaluate the supervised learning model is Area Under the Receiver Operating Characteristic Curve (short **ROC AUC,** see https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc, https://towardsdatascience.com/an-understandable-guide-to-roc-curves-and-auc-and-why-and-when-to-use-them-92020bc4c5c1) because we have a very imbalanced binary classification problem to solve.

Analysis

Data Exploration

The given data consists of four datasets and 2 metadata files describing the features of the dataset.

Filename	Description	Records (rows)	Features (columns)	Labels
Udacity_AZDIAS_052018.csv:	Demographics data for the general population of Germany	891,211	366	
Udacity_CUSTOMERS_052018.csv:	Demographics data for customers of a mail-order company 3 additional columns	191,652	369	
Udacity_MAILOUT_052018_TRAIN.csv	Demographics data for individuals who were targets of a marketing campaign	42,982	367	1
Udacity_MAILOUT_052018_TEST.csv	Demographics data for individuals who were targets of a marketing campaign	42,833	366	

Metadata

- DIAS Information Levels - Attributes 2017.xlsx: is a top-level list of attributes and descriptions, organized by informational category.
- DIAS Attributes - Values 2017.xlsx: is a detailed mapping of data values for each feature in alphabetical order.
- The metadata file does not describe all features used in the dataset. Features for that no description exist are

  'AKT_DAT_KL', 'ALTERSKATEGORIE_FEIN', 'ALTER_KIND1', 'ALTER_KIND2', 'ALTER_KIND3', 'ALTER_KIND4', 'ANZ_KINDER', 'ANZ_STATISTISCHE_HAUSHALTE', 'ARBEIT', 'CAMEO_INTL_2015', 'CJT_KATALOGNUTZER', 'CJT_TYP_1', 'CJT_TYP_2', 'CJT_TYP_3', 'CJT_TYP_4', 'CJT_TYP_5', 'CJT_TYP_6', 'D19_BUCH_CD', 'D19_KONSUMTYP_MAX',

D19_LETZTER_KAUF_BRANCHE', 'D19_SOZIALES', 'D19_TELKO_ONLINE_QUOTE_12', 'D19_VERSI_DATUM', 'D19_VERSI_OFFLINE_DATUM', 'D19_VERSI_ONLINE_DATUM', 'D19_VERSI_ONLINE_QUOTE_12', 'DSL_FLAG', 'EINGEFUEGT_AM', 'EINGEZOGENAM_HH_JAHR', 'EXTSEL992', 'FIRMENDICHTE', 'GEMEINDETYP', 'HH_DELTA_FLAG', 'KBA13_ANTG1', 'KBA13_ANTG2', 'KBA13_ANTG3', 'KBA13_ANTG4', 'KBA13_BAUMAX', 'KBA13_CCM_1401_2500', 'KBA13_GBZ', 'KBA13_HHZ', 'KBA13_KMH_210', 'KK_KUNDENTYP', 'KOMBIALTER', 'KONSUMZELLE', 'MOBI_RASTER', 'RT_KEIN_ANREIZ', 'RT_SCHNAEPPCHEN', 'RT_UEBERGROESSE', 'SOHO_KZ', 'STRUKTURTYP', 'UMFELD_ALT', 'UMFELD_JUNG', 'UNGLEICHENN_FLAG', 'VERDICHTUNGSRAUM', 'VHA', 'VHN', 'VK_DHT4A', 'VK_DISTANZ', 'VK_ZG11'

Index (Unique Identifier)

The “LNR” column can be used as index for the datasets.

Variable Value Checks

We now check which values occur in the data set and if they match with the values defined in the metadata files.

A first glance on the used values shows that the dataset contains huge number of floats also the metadata just defines int values and some string codes

```
array(['-1', '0', '1', '2', '3', '-1, 0', '4', '9', '5', '6', '7', '8',
       '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20',
       '21', '…', '1A', '1B', '1C', '1D', '1E', '2A', '2B', '2C', '2D',
       '3A', '3B', '3C', '3D', '4A', '4B', '4C', '4D', '4E', '5A', '5B',
       '5C', '5D', '5E', '5F', '6A', '6B', '6C', '6D', '6E', '6F', '7A',
       '7B', '7C', '7D', '7E', '8A', '8B', '8C', '8D', '9A', '9B', '9C',
       '9D', '9E', '22', '23', '24', '25', '31', '32', '33', '34', '35',
       '41', '42', '43', '44', '45', '51', '52', '53', '54', '55',
       '-1, 9', '26', '27', '28', '29', '30', '36', '37', '38', '39',
       '40', 'O', 'W'], dtype=object)
```

Meaning of Value “…”

	Attribute	Description	Value	Meaning
36	ANZ_HAUSHALTE_AKTIV	number of households in the building	...	numeric value (typically coded from 1-10)
37	ANZ_HH_TITEL	number of academic title holder in building	...	numeric value (typically coded from 1-10)
38	ANZ_PERSONEN	number of adult persons in the household	...	numeric value (typically coded from 1-3)
39	ANZ_TITEL	number of professional title holder in household	...	numeric value (typically coded from 1-10)
711	GEBURTSJAHR	year of birth	...	numeric value
1167	KBA13_ANZAHL_PKW	number of cars in the PLZ8	...	numeric value
1986	MIN_GEBAEUDEJAHR	year the building was first mentioned in our d...	...	numeric value

A check of the data set show that there are many values that are not defined in the metadata value set. Let’s identify which values that are and which columns (features) use them.

- E.g. ANZ_HAUSHALTE_AKTIV,ANZ_STATISTISCHE_HAUSHALTE, KBA13_ANZAHL_PKW use int and floats from 0 to N -> obviously that is the total number of items counted.
- D19_LETZTER_KAUF_BRANCHE contains categorical values like 'D19_LEBENSMITTEL', 'D19_BANKEN_DIREKT', 'D19_BANKEN_GROSS'
- EINGEZOGENAM_HH_JAHR GEBURTSJAHR MIN_GEBAEUDEJAHR are columns of type date (year)

Unknown Data

The features of the datasets are mainly numerical encoded nominal and ordinal values, e.g.

	Attribute	Description	Value	Meaning
0	AGER_TYP	best-ager typology	-1	unknown
1	AGER_TYP	best-ager typology	0	no classification possible
2	AGER_TYP	best-ager typology	1	passive elderly
3	AGER_TYP	best-ager typology	2	cultural elderly
4	AGER_TYP	best-ager typology	3	experience-driven elderly
5	ALTERSKATEGORIE_GROB	age classification through prename analysis	-1, 0	unknown
6	ALTERSKATEGORIE_GROB	age classification through prename analysis	1	< 30 years
7	ALTERSKATEGORIE_GROB	age classification through prename analysis	2	30 - 45 years
8	ALTERSKATEGORIE_GROB	age classification through prename analysis	3	46 - 60 years
9	ALTERSKATEGORIE_GROB	age classification through prename analysis	4	> 60 years
10	ALTERSKATEGORIE_GROB	age classification through prename analysis	9	uniformly distributed

As you can see there are encoded values for “unknown”. These values are handled in the same way as missing values (NaNs / NULLs). That means I replace unknown and missing values by a measure of central tendency. For ordinal values the mode and mean are useful replacements for nominal just mode (https://en.wikipedia.org/wiki/Mode_(statistics)).

Duplicate data

Let’s check if the data sets contain records for the same individual.

check for duplicates

check if dataset contains duplicate records based on column ID LNR

```
df_azdias.index.duplicated().sum()
```

✓ 1.6s

```
0
```

```
df_customers.index.duplicated().sum()
```

✓ 0.4s

```
0
```

There no records for the same individual. Next we check if there are same data points that means if there exist exactly the same records ignoring LNR column.

```
t=df_azdias.drop_duplicates()
df_azdias.shape[0] -t.shape[0]
```
✓ 161.4s

```
45762
```

```
t=df_customers.drop_duplicates()
df_customers.shape[0] -t.shape[0]
```
✓ 8.4s

```
41121
```

This shows that the general population and the customer dataset have little more than 40,000 identical records. That are ~5% for the general population and ~21% for the customer dataset

Handling of Data Load Errors

During the load of the datasets we face some warning that columns 18 and 19 have mixed types. These columns are CAMEO_DEUG_2015, CAMEO_INTL_2015.

A check of these two columns shows that they contain numerical encoded ordinal values but also strings “X” and “XX”

```
# column 18 has 0-based index 17
df_azdias.iloc[:,17].unique()
```

```
array(['8', '3', '6', '5', '9', nan, 6.0, 2.0, 8.0, '4', '2', 3.0, 4.0,
       9.0, '7', 1.0, 7.0, 5.0, '1', 'X'], dtype=object)
```

```
# column 19 has 0-based index 18
df_customers.iloc[:,18].unique()
```

```
array(['24', '33', nan, '23', '13', '55', '43', 24.0, '15', '25', '41',
       '22', 15.0, '14', '32', 41.0, 31.0, 51.0, '35', 54.0, 14.0, '54',
       '52', 13.0, 22.0, '34', '31', 44.0, '12', 32.0, 25.0, 43.0, 12.0,
       '44', '51', '45', 35.0, 45.0, 34.0, 23.0, 'XX', 55.0, 33.0, 52.0],
      dtype=object)
```

We replace them by np.NaN.

Analyse Feature Values

A check of the values used in the feature columns show that many of them contain a mix of int and float values. In addition there are columns with a wide range of different values. We see that there are date columns like EINGEZOGENAM_HH_JAHR, GEBURTSJAHR and MIN_GEBAEUDEJAHR.

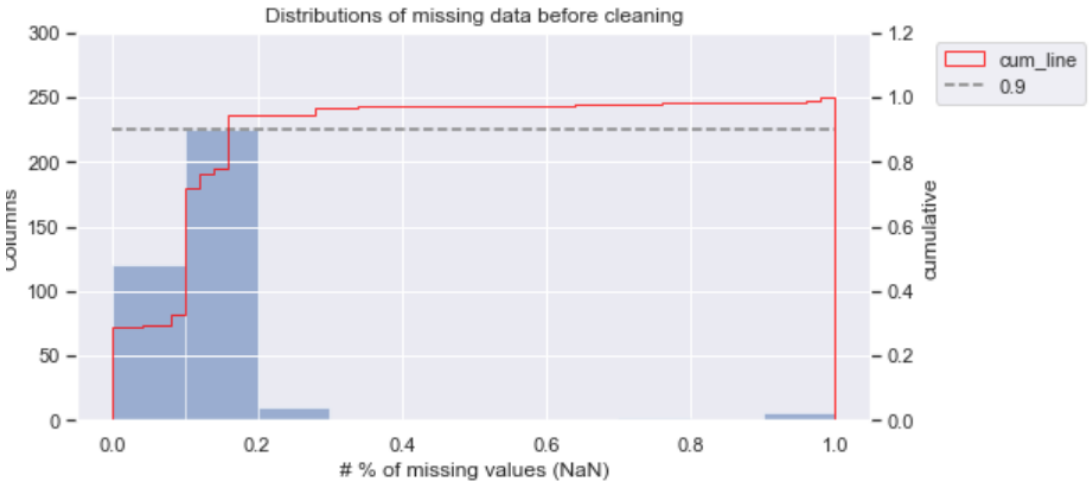
These columns are converted to int.

Missing Data

Let's now check how many data is missing.

Note that we treat unknown values as missing values

A first check shows that for most columns less than 20% of data is missing. For a few columns almost all data is missing

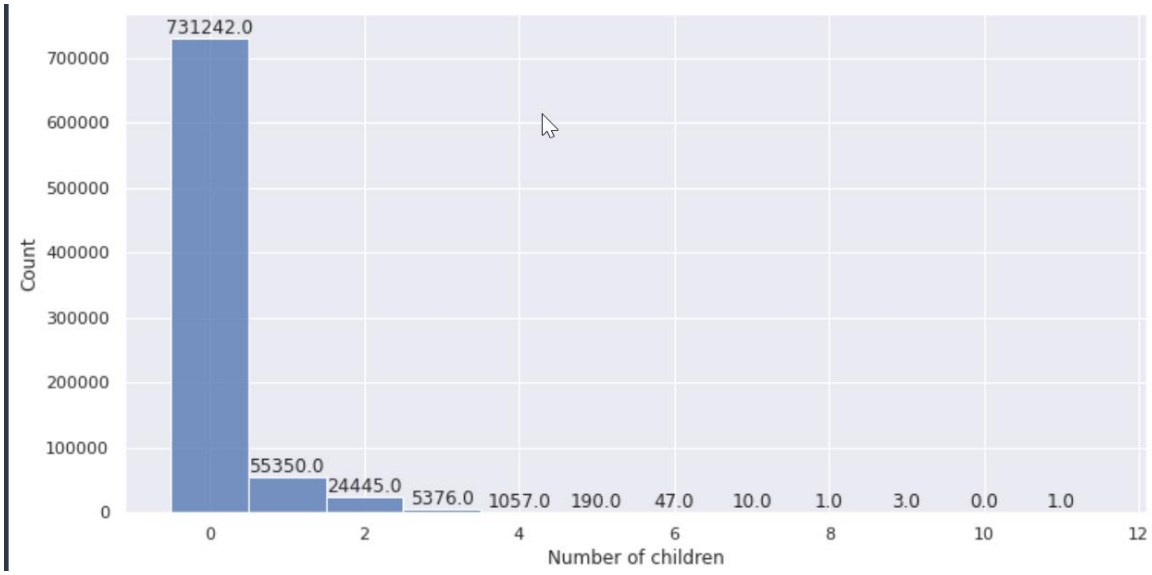


The following table shows that there are 10 columns for that more than 30% of data is missing. Five of them are the ALTER_KIND and TITEL_KZ columns for that more than >90% is missing. For all other columns less than 30% is missing.

	abs	pct
ALTER_KIND4	890016	0.998648
TITEL_KZ	889061	0.997576
ALTER_KIND3	885051	0.993077
ALTER_KIND2	861722	0.966900
ALTER_KIND1	810163	0.909048
AGER_TYP	677503	0.760196
EXTSEL992	654153	0.733996
KK_KUNDENTYP	584612	0.655967
KBA05_BAUMAX	476524	0.534687
ALTER_HH	310267	0.348137
ALTERSKATEGORIE_FEIN	262947	0.295041
D19_LETZTER_KAUF_BRANCHE	257113	0.288495
D19_GESAMT_ONLINE_QUOTE_12	257113	0.288495
D19_KONSUMTYP	257113	0.288495
D19_VERSI_ONLINE_QUOTE_12	257113	0.288495
D19_LOTTO	257113	0.288495
D19_SOZIALES	257113	0.288495
D19_BANKEN_ONLINE_QUOTE_12	257113	0.288495
D19_TELKO_ONLINE_QUOTE_12	257113	0.288495
D19_VERSAND_ONLINE_QUOTE_12	257113	0.288495

Therefore we will drop the 10 columns with more than 30% missing data and keep the others. In addition we drop D19_LETZTER_KAUF_BRANCHE as this columns contains >30% values "D19_UNBEKANNT" which means also unknown.

Note: The ALTER Kind columns have a huge number of missing values as they indicate the age of child 1..4 and if values are available or missing is directly related to ANZ_KIND which indicated the number of children.



Categorical Features

Most features of the dataset contain categorical data but it is already encoded by int values. But some features still have categorical data. In order to be able to apply machine learning algorithms we need to encode them (See also https://machinelearningmastery.com/how-to-prepare-categorical-data-for-deep-learning-in-python/). The features that are not encoded categorical are

```
df_azdias_cleaned.select_dtypes(include='object').head()
```

LNR	CAMEO_DEU_2015	CAMEO_DEUG_2015	CAMEO_INTL_2015	D19_LETZTER_KAUF_BRANCHE	EINGEFUEGT_AM	OST_WEST_KZ
393872	8B	8	41	NaN	1992-02-10 00:00:00	W
502515	3D	3	25	D19_UNBEKANNT	1992-02-10 00:00:00	W
783105	6B	6	43	D19_SONSTIGE	2007-03-07 00:00:00	W
695574	5D	5	34	NaN	1992-02-10 00:00:00	W
456575	9D	9	51	NaN	1997-11-08 00:00:00	W

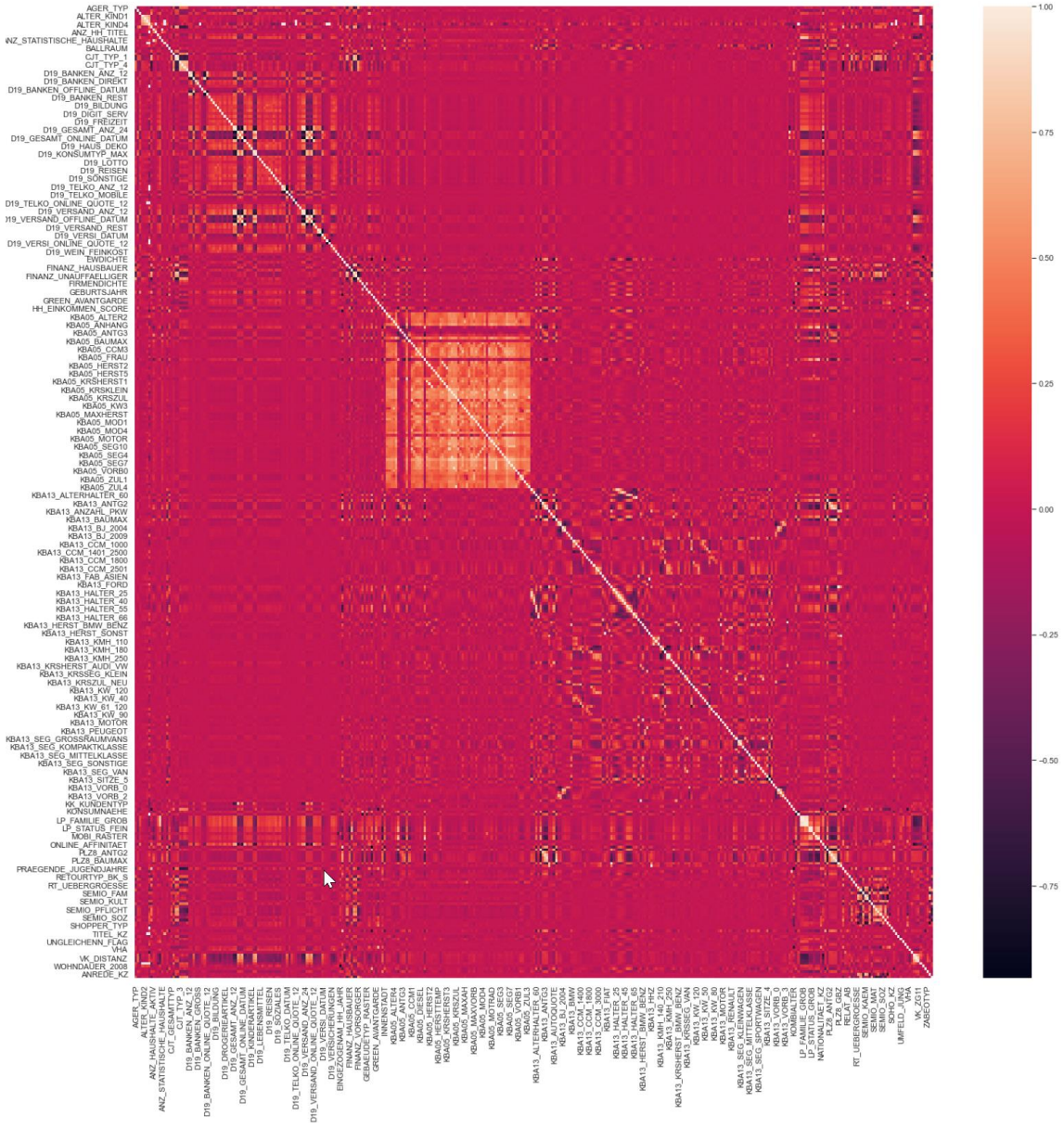
We will apply the following actions

variable	type	action
CAMEO_DEU_2015	nominal	drop - this column seems to be based on not real measurable feature
CAMEO_DEUG_2015	nominal	encoded categorical variable - contains invalid strings 'X'
CAMEO_INTL_2015	nominal	encoded categorical variable - contains invalid strings 'XX'
D19_LETZTER_KAUF_BRANCHE	nominal	replace by one hot encoding
EINGEFUEGT_AM	date	drop - this is just the date when the record has been added
OST_WEST_KZ	nominal	replace by binary 0 and 1

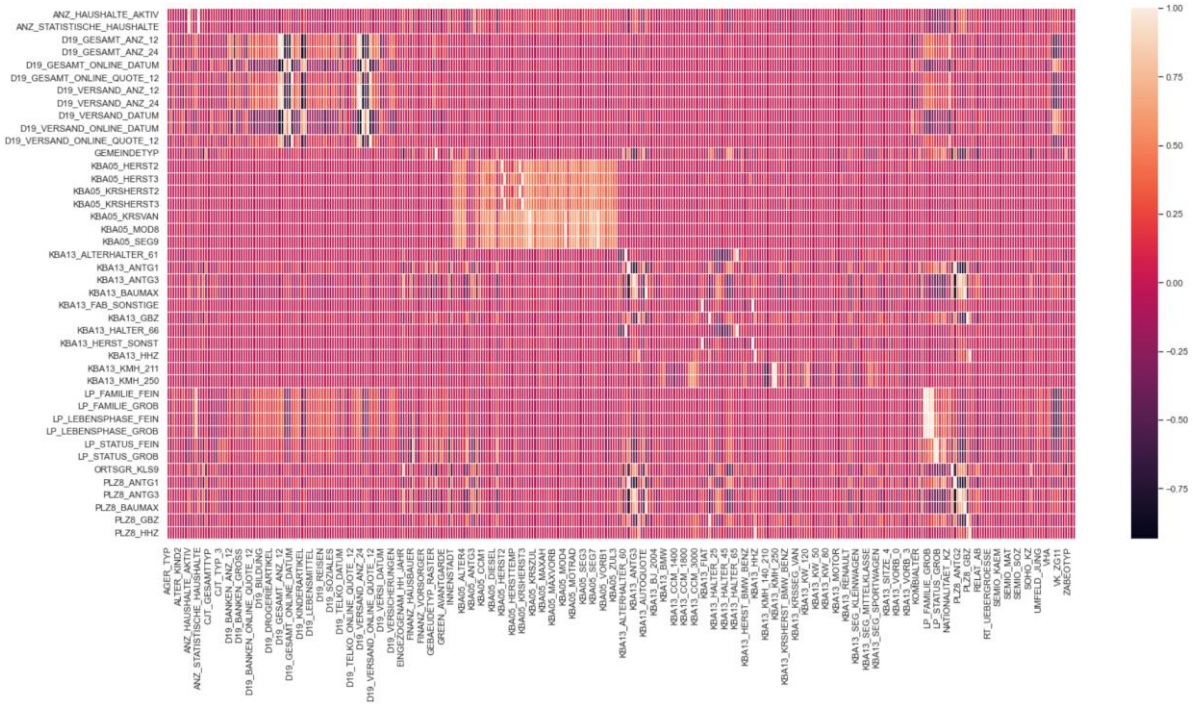
Data Correlation

Now let's have a look on the correlation of the data set.

The heat map shows some areas that are heavily positive correlated. Let's have a closer look on these main correlations



The highest absolute correlation (>0.9) have these features




```
{'ANZ_HAUSHALTE_AKTIV': [{'ANZ_STATISTISCHE_HAUSHALTE': 0.9825454972131037}],
 'D19_GESAMT_ANZ_12': [{'D19_GESAMT_ANZ_24': 0.9059313780609212},
                       {'D19_VERSAND_ANZ_12': 0.9057561799122893}],
 'D19_GESAMT_ANZ_24': [{'D19_VERSAND_ANZ_24': 0.9209304397170982}],
 'D19_GESAMT_ONLINE_DATUM': [{'D19_VERSAND_ONLINE_DATUM': 0.9439061906732233},
                             {'D19_VERSAND_DATUM': 0.9130349398350104}],
 'D19_GESAMT_ONLINE_QUOTE_12': [{'D19_VERSAND_ONLINE_QUOTE_12': 0.9166732854362937}],
 'D19_VERSAND_ANZ_12': [{'D19_VERSAND_ANZ_24': 0.9079456507492998}],
 'D19_VERSAND_DATUM': [{'D19_VERSAND_ONLINE_DATUM': 0.9571270796413961}],
 'KBA05_HERST2': [{'KBA05_KRSHERST2': 0.9038264918739526}],
 'KBA05_HERST3': [{'KBA05_KRSHERST3': 0.9250961625474592}],
 'KBA05_KRSVAN': [{'KBA05_SEG9': 0.9034826098405083}],
 'KBA05_MOD8': [{'KBA05_SEG9': 0.90329170121744}],
 'KBA13_ALTERHALTER_61': [{'KBA13_HALTER_66': 0.9295604291892366}],
 'KBA13_ANTG1': [{'PLZ8_ANTG1': 0.917660500203741}],
 'KBA13_ANTG3': [{'PLZ8_ANTG3': 0.9109969171296487}],
 'KBA13_BAUMAX': [{'PLZ8_BAUMAX': 0.950169749558509}],
 'KBA13_FAB_SONSTIGE': [{'KBA13_HERST_SONST': 1.0}],
 'KBA13_GBZ': [{'PLZ8_GBZ': 0.9793321750444082}],
 'KBA13_HHZ': [{'PLZ8_HHZ': 0.9679790215095707}],
 'KBA13_KMH_211': [{'KBA13_KMH_250': 0.9583967544005739}],
 'LP_FAMILIE_FEIN': [{'LP_FAMILIE_GROB': 0.9837911261774264},
                     {'LP_LEBENSPHASE_GROB': 0.9412334123749199},
                     {'LP_LEBENSPHASE_FEIN': 0.915689675351284}],
 'LP_LEBENSPHASE_FEIN': [{'LP_LEBENSPHASE_GROB': 0.9898953640647802},
                         {'LP_FAMILIE_GROB': 0.9402328586954048}],
 'LP_STATUS_FEIN': [{'LP_STATUS_GROB': 0.9823887737174377}]}
```

Detailed view on some selected very high correlated features:

Methodology

Data Pre-processing

In order to prepare the data for the further analysis we will execute the following pre-processing steps:

1. Drop additional customer data set
   We drop the columns ‘CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP' as these are just contained in the customer dataset

2. Handle data load errors
   We convert “X” and “XX” values in columns ‘CAMEO_DEUG_2015', 'CAMEO_INTL_2015’ to np.NaN

3. Drop duplicates
   Drop identical rows.

4. Handle unknown values
   We replace unknown values with np.NaN in order to treat them like missing data. We use the Metadata to identify unknown values. We also convert all negative values to np.NaN as we assume that this is also unknown as -1 is in general for unknown according to the metadata.

5. Encode categorical variables
   Encoding of categorical variables is not required as the only not encoded categorical features are dropped.

variable	type	action
CAMEO_DEU_2015	nominal	drop - this column seems to be based on not real measurable feature
CAMEO_DEUG_2015	nominal	encoded categorical variable - contains invalid strings 'X'
CAMEO_INTL_2015	nominal	encoded categorical variable - contains invalid strings 'XX'
D19_LETZTER_KAUF_BRANCHE	nominal	replace by one hot encoding
EINGEFUEGT_AM	date	drop - this is just the date when the record has been added
OST_WEST_KZ	nominal	replace by binary 0 and 1

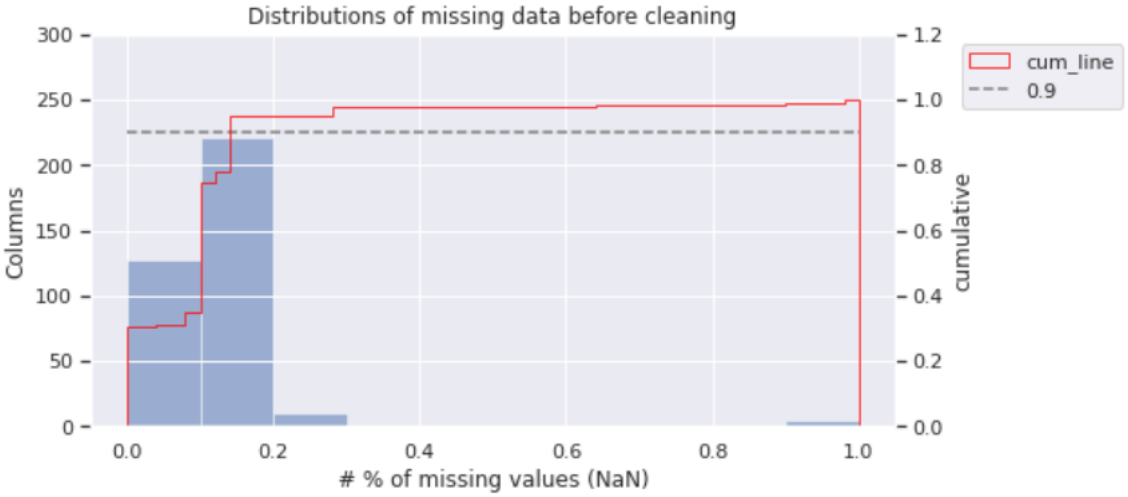
6. Drop columns
   a. Too many missing values
      Drop all columns with more than 30% missing data. That are:

      'ALTER_KIND4', 'ALTER_KIND3', 'ALTER_KIND2', 'ALTER_KIND1',
      'ALTER_HH', 'AGER_TYP',
      'EXTSEL992', 'KK_KUNDENTYP', 'KBA05_BAUMAX', 'TITEL_KZ'

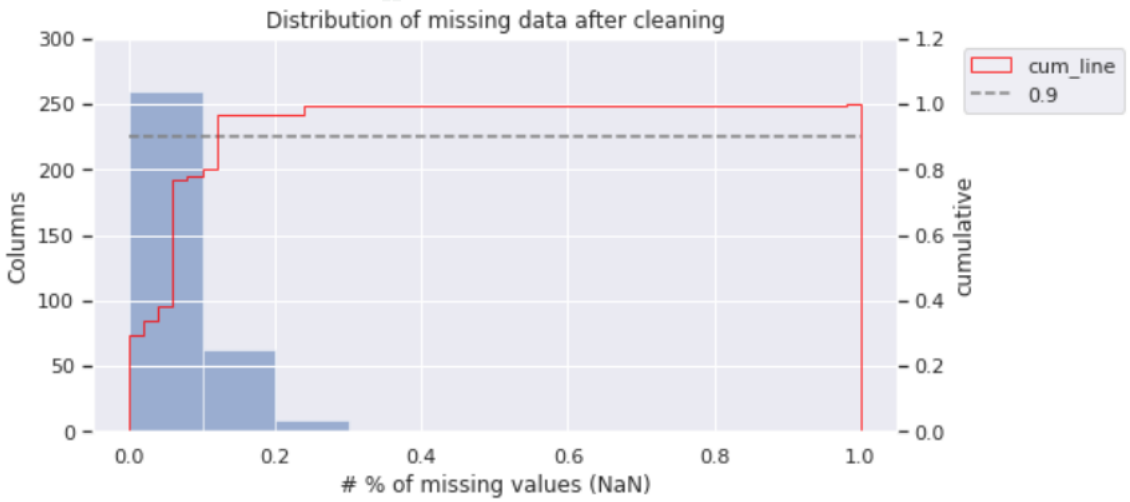
   b. Too much correlation
      Drop columns with abs(correlation) > 0.9. That are
      'CAMEO_DEU_2015','LP_STATUS_GROB','LP_FAMILIE_GROB','D19_VERSAND_ANZ_24','LP_LEBENSPHASE_FEIN', 'ANZ_STATISTISCHE_HAUSHALTE', 'CAMEO_INTL_2015',
      'D19_VERSAND_ONLINE_DATUM', 'KBA13_HALTER_66', 'KBA13_HERST_SONST', ,
      'PLZ8_BAUMAX', 'PLZ8_GBZ', 'PLZ8_HHZ', 'D19_GESAMT_ANZ_24',
      'D19_VERSAND_ANZ_12', 'D19_VERSAND_DATUM', 'KBA05_KRSHERST2',
      'KBA05_KRSHERST3', 'KBA05_SEG9', 'KBA13_KMH_211', 'PLZ8_ANTG1',
      'PLZ8_ANTG3'

Comparing the data before and after the cleaning

Missing Values



After cleaning

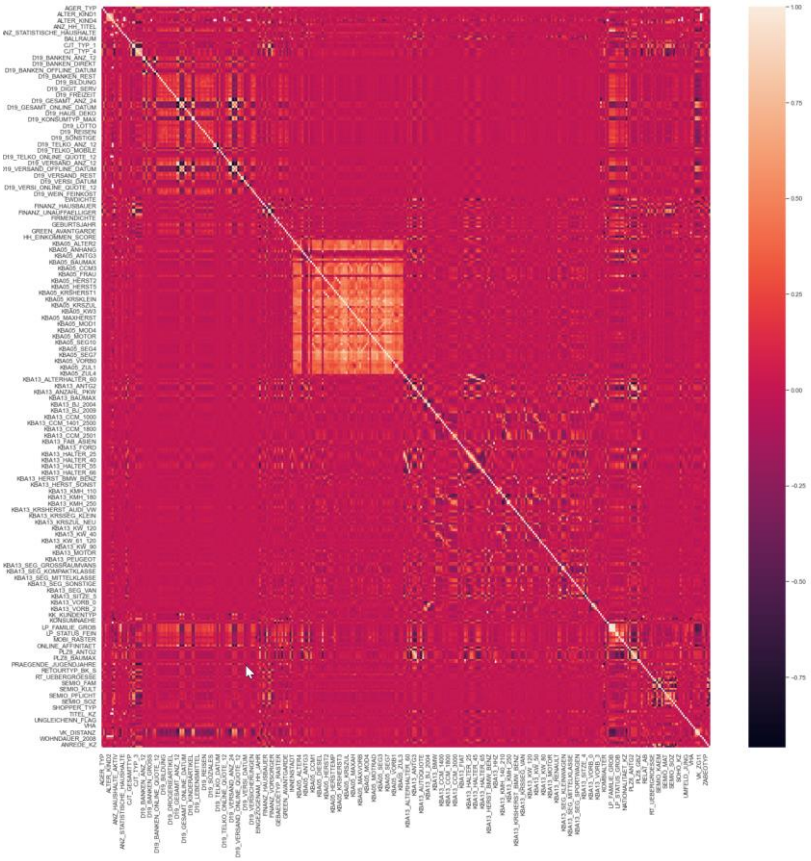


There is a significant increase of columns with less missing data. The main reason are:

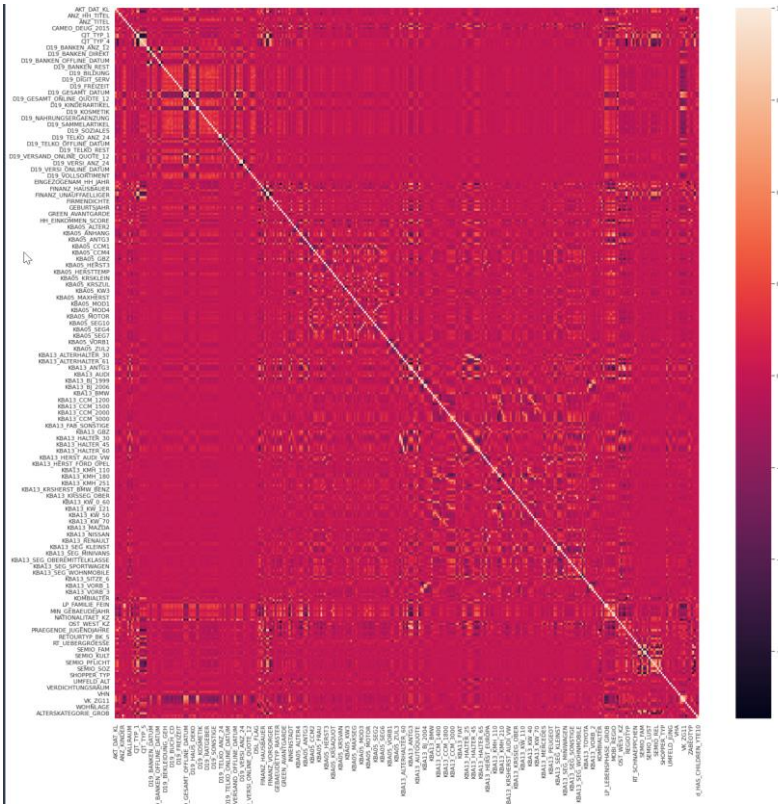
- Dropping of columns with more than 30% missing data
- some "unknown" values were replaced values by np.NaN
- Duplicate rows have been removed. There are for many columns the same number of removed missing values ~45730. This is the cause the decrease of number of columns for bin 2 (10-20%) and increase of bin 1 (0% - 10%)

Correlation

Before cleaning



After cleaning



You can see that the very high correlated data areas are removed after the cleaning.

Implementation

The implementation consists of two step. First part is the unsupervised learning and the second the supervised learning.

Unsupervised learning

In order to segment the customers with unsupervised learning we use a clustering algorithm.
For clustering there is number of popular algorithms. For the algorithm selection we focus on the ones that scikit-learn provides and the article clustering algorithms with python (https://machinelearningmastery.com/clustering-algorithms-with-python/) or How to Combine PCA and K-means Clustering (https://365datascience.com/tutorials/python-tutorials/pca-k-means/) describe.

According to the references the most popular algorithms are

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

The bold one are the one(s) we use.

The dimension of the dataset is quite high so that it is worth to consider a reduction of the dimensionality which will increase the performance and in many cases the accuracy of algorithm. In particular the popular K-means which we will use will profit from it (See e.g. PCA with k-means https://365datascience.com/tutorials/python-tutorials/pca-k-means/)

The main steps outlined in the combination of all guides are

The approach is

1. **Load and Prepare Data**

2. **PCA - Principal Component Analysis**
   This algorithms is also provided by scikit-learn. It will transform the given space of features to new space with basis vectors that are linear combinations of the given features so that the new vectors point in direction of the maximum variance. For more information how to execute and interpret the values see [In Depth: Principal Component Analysis](https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html)
   a. Impute missing data
   b. Standardize data
   c. Execute PCA

d. Define the level of explained variance to be kept -> this defines number of components to keep
e. Train (FIT) PCA on reduced number of components

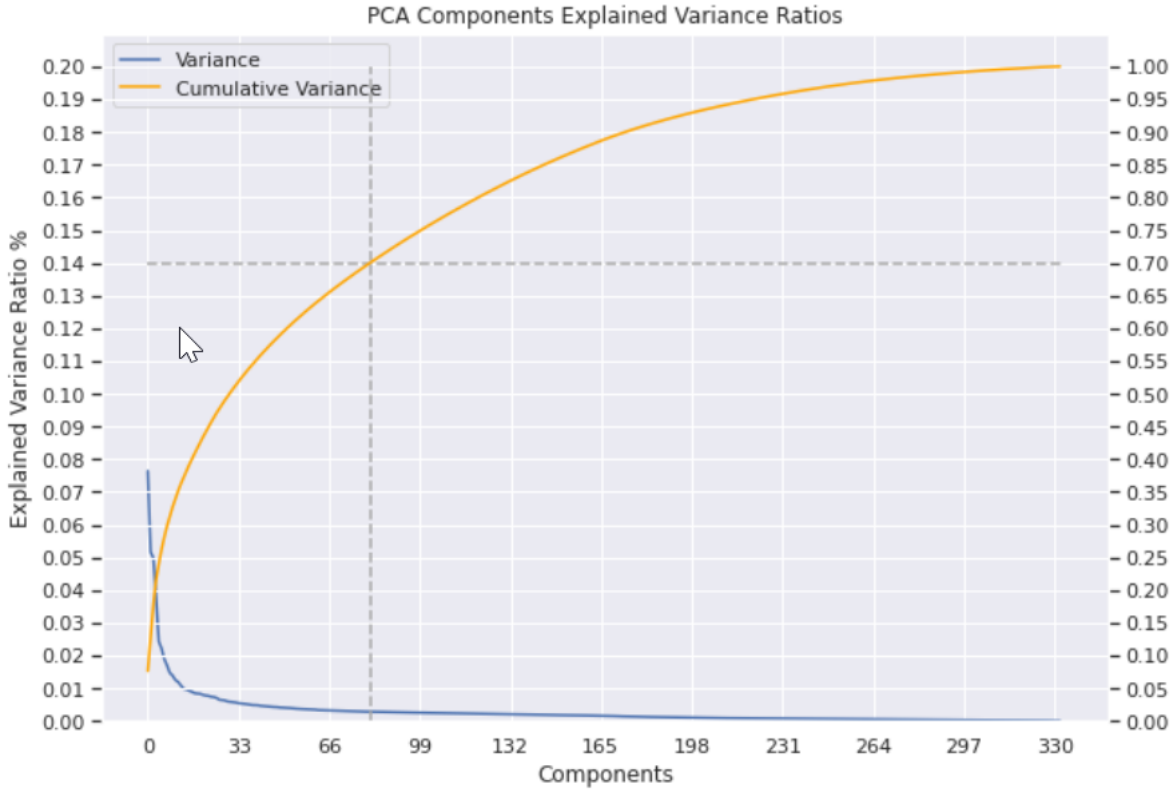
3. **K-means**
   Finally we cluster the data by applying KMeans on the data. We use 3 steps for this.
   a. Fit KMeans on the general population data
   b. Determine the best number of clusters
   c. Cluster general and customer data on defined number of clusters

For the complete process we will use a sklearn pipeline to chain the steps

PCA – Principal Component Analysis

Executing the PCA and plotting the Explained variance over the number of components show the following graph.

7



The dashed line shows that 70% of the variance is explained by 81 components.

Let's now check how much variance is explained by the first components and which features are the driving ones for these ones.

Component- 1

Component I

```
pca_processor.get_component_features(0)[:n_show]
```

```
Explained Variance in % for cmponent: 0: 0.0764
```

	Attribute	Variance	Description
0	MOBI_REGIO	0.141682	moving patterns
1	LP_STATUS_FEIN	0.138742	social status fine
2	KBA05_ANTG1	0.130090	number of 1-2 family houses in the cell
3	KBA13_AUTOQUOTE	0.125690	share of cars per household within the PLZ8
4	KBA05_AUTOQUOT	0.125378	share of cars per household
5	KBA05_GBZ	0.123855	number of buildings in the microcell
6	FINANZ_MINIMALIST	0.119042	financial typology: low financial interest

```
pca_processor.get_component_features(0)[-n_show:]
```

```
Explained Variance in % for cmponent: 0: 0.0764
```

	Attribute	Variance	Description
309	FINANZ_HAUSBAUER	-0.109414	financial typology: main focus is the own house
310	ORTSGR_KLS9	-0.114175	size of the community
311	ORTSGR_KLS9	-0.114175	'- classified number of inhabitants
312	EWDICHTE	-0.116134	density of inhabitants per square kilometer
313	HH_EINKOMMEN_SCORE	-0.127086	estimated household net income
314	PLZ8_ANTG4	-0.130267	number of >10 family houses in the PLZ8
315	CAMEO_DEUG_2015	-0.131404	CAMEO classification 2015 - Uppergroup

Component 2

Component II

```
pca_processor.get_component_features(1)[:n_show]
```

```
Explained Variance in % for cmponent: 1: 0.0517
```

	Attribute	Variance	Description
0	PRAEGENDE_JUGENDJAHRE	0.177636	dominating movement in the person's youth (ava...
1	FINANZ_SPARER	0.171276	financial typology: money saver
2	ONLINE_AFFINITAET	0.154477	online affinity
3	FINANZ_ANLEGER	0.151385	financial typology: investor
4	SEMIO_PFLICHT	0.140380	affinity indicating in what way the person is ...
5	SEMIO_TRADV	0.138184	affinity indicating in what way the person is ...
6	SEMIO_RAT	0.127204	affinity indicating in what way the person is ...

```
pca_processor.get_component_features(1)[-n_show:]
```

```
Explained Variance in % for cmponent: 1: 0.0517
```

	Attribute	Variance	Description
309	W_KEIT_KIND_HH	-0.108596	likelihood of a child present in this household
310	D19_GESAMT_DATUM	-0.109094	actuality of the last transaction with the com...
311	D19_GESAMT_ONLINE_DATUM	-0.116738	actuality of the last transaction with the com...
312	RETOURTYP_BK_S	-0.116840	return type
313	SEMIO_LUST	-0.118793	affinity indicating in what way the person is ...
314	ALTERSKATEGORIE_GROB	-0.148994	age classification through prename analysis
315	FINANZ_VORSORGER	-0.168108	financial typology: be prepared

Component 3

Component III

```
pca_processor.get_component_features(2)[:n_show]
```

```
Explained Variance in % for cmponent: 2: 0.0499
```

	Attribute	Variance	Description
0	KBA13_HERST_BMW_BENZ	0.183161	share of BMW & Mercedes Benz within the PLZ8
1	KBA13_SEG_OBEREMITTELKLASSE	0.158405	share of upper middle class cars and upper cla...
2	KBA13_MERCEDES	0.156953	share of MERCEDES within the PLZ8
3	KBA13_BMW	0.152231	share of BMW within the PLZ8
4	KBA13_SITZE_4	0.147854	number of cars with less than 5 seats in the PLZ8
5	KBA13_SEG_SPORTWAGEN	0.139257	share of sportscars within the PLZ8
6	KBA05_HERST1	0.121165	share of top German manufacturer (Mercedes, BMW)

```
pca_processor.get_component_features(2)[-n_show:]
```

```
Explained Variance in % for cmponent: 2: 0.0499
```

	Attribute	Variance	Description
309	OST_WEST_KZ	-0.111558	flag indicating the former GDR/FRG
310	KBA13_KMH_180	-0.114524	share of cars with max speed between 110 km/h ...
311	KBA13_SEG_KOMPAKTKLASSE	-0.117896	share of lowe midclass cars (Ford Focus etc.) ...
312	KBA13_HALTER_25	-0.121013	share of car owners between 21 and 25 within t...
313	KBA13_KMH_140_210	-0.125558	share of cars with max speed between 140 and 2...
314	KBA13_SEG_KLEINWAGEN	-0.132128	share of small and very small cars (Ford Fiest...
315	KBA13_SITZE_5	-0.153357	number of cars with 5 seats in the PLZ8

Unsupervised Learning - Customer Segmentation (Clustering)

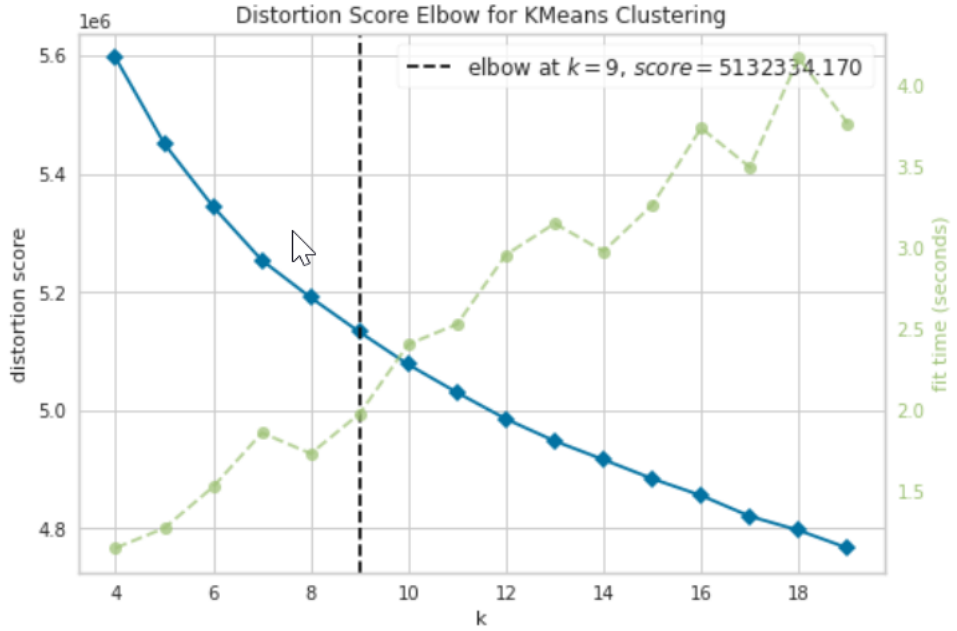
As mentioned in the section introduction we focus on the KMeans algorithm for unsupervised clustering.

Determine number of Clusters

In order to find the best number of clusters we use the yellowbricks library that applies the elbow method that is also referenced in the guides mentioned at the beginning of this chapter (https://en.wikipedia.org/wiki/Elbow_method_(clustering)).

After we have determined the number of clusters we will apply PCA on cleaned customer dataset and run then the KMeans algorithm on customer and general population dataset for the given number of clusters.

Finally we can compare the proportion per cluster between customer and general population.



The elbow is at k=9 clusters. As this is not a clear elbow we will compare the segmentation for different number n =[8,…14] of clusters.

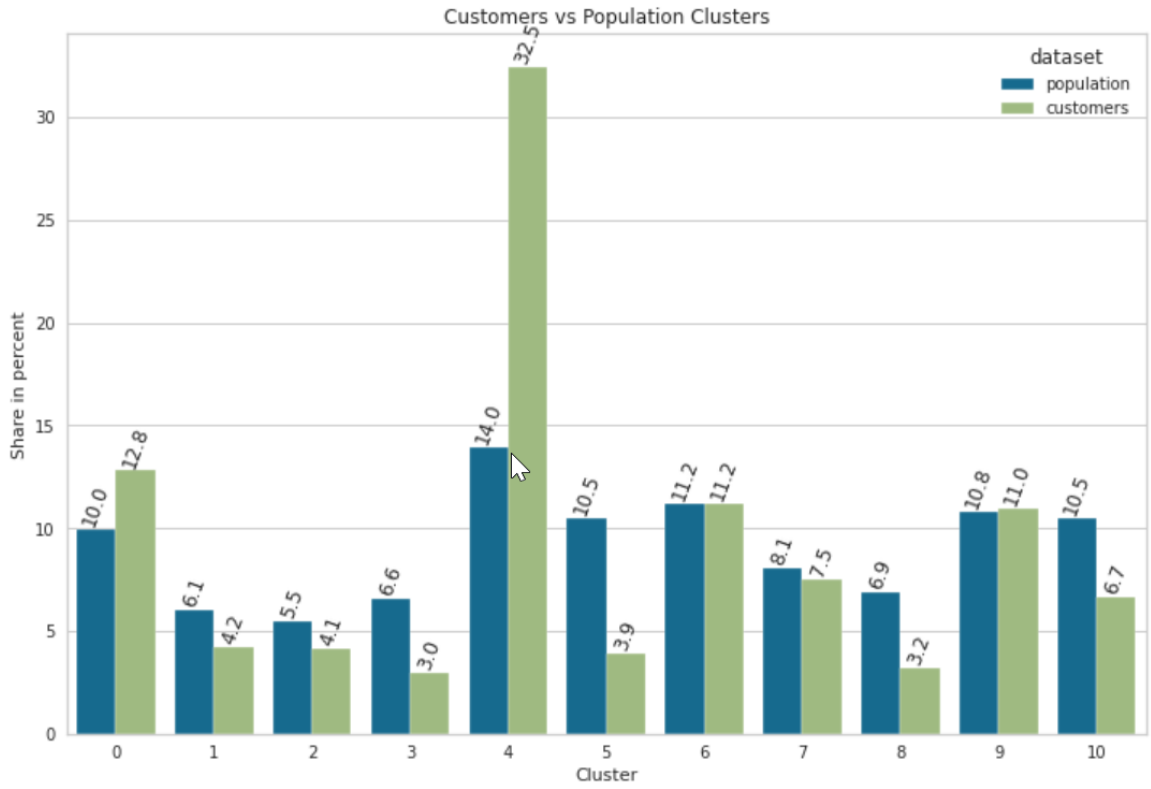
For n=10,11,12 clusters the result are listed in table form below

dataset cluster	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
3	27.51	14.10	27.51	14.10	13.41	13.41
5	14.11	16.55	41.63	30.65	10.98	-2.43
2	12.77	10.45	54.39	41.10	13.29	2.31
9	9.62	7.68	64.01	48.78	15.23	1.94
7	9.34	9.85	73.35	58.63	14.72	-0.51
8	8.60	9.21	81.95	67.84	14.11	-0.61
6	6.65	7.40	88.60	75.24	13.37	-0.75
1	5.91	8.02	94.51	83.26	11.26	-2.11
0	3.09	10.15	97.61	93.41	4.20	-7.06
4	2.39	6.59	100.00	100.00	-0.00	-4.20

dataset cluster	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
4	32.47	13.95	32.47	13.95	18.52	18.52
0	12.80	9.95	45.27	23.90	21.36	2.85
6	11.20	11.21	56.46	35.11	21.35	-0.01
9	10.96	10.80	67.42	45.92	21.50	0.15
7	7.51	8.08	74.93	54.00	20.93	-0.57
10	6.67	10.53	81.60	64.52	17.07	-3.86
1	4.20	6.05	85.80	70.58	15.22	-1.85
2	4.14	5.47	89.94	76.05	13.89	-1.33
5	3.90	10.51	93.84	86.56	7.28	-6.61
8	3.19	6.89	97.02	93.45	3.58	-3.70
3	2.98	6.55	100.00	100.00	-0.00	-3.58

dataset cluster	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
11	28.44	11.61	28.44	11.61	16.83	16.83
5	13.04	11.60	41.48	23.21	18.27	1.44
10	11.99	8.88	53.47	32.09	21.38	3.11
7	9.35	9.65	62.83	41.74	21.09	-0.30
2	8.99	7.86	71.81	49.60	22.21	1.13
8	6.42	7.67	78.23	57.27	20.96	-1.25
1	5.04	6.85	83.27	64.12	19.15	-1.81
6	4.80	6.10	88.07	70.22	17.85	-1.30
9	4.48	5.59	92.55	75.81	16.74	-1.11
3	3.01	10.29	95.56	86.10	9.46	-7.28
0	2.64	8.41	98.20	94.50	3.70	-5.76
4	1.80	5.50	100.00	100.00	0.00	-3.70

The one in the middle (clusters = 11) seems to be best suited as the differences between customer and general population in the clusters is best. In the first 4 clusters is more than 67% of all customer data whereas just 46% of general population is in these four.



As you can see there are much more customers in cluster 4 compared to general population (~18.5% difference). Also in cluster 0, the customer proportion is almost 3% higher. In these 2 clusters are more than 45% of the customers compared to ~24% of general population.

If we take the first 4 clusters ordered by customers_pct we have more than 67% of all customers included which is 21.5% compared to general population.

dataset cluster	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
4	32.47	13.95	32.47	13.95	18.52	18.52
0	12.80	9.95	45.27	23.90	21.36	2.85
6	11.20	11.21	56.46	35.11	21.35	-0.01
9	10.96	10.80	67.42	45.92	21.50	0.15
7	7.51	8.08	74.93	54.00	20.93	-0.57
10	6.67	10.53	81.60	64.52	17.07	-3.86
1	4.20	6.05	85.80	70.58	15.22	-1.85
2	4.14	5.47	89.94	76.05	13.89	-1.33
5	3.90	10.51	93.84	86.56	7.28	-6.61
8	3.19	6.89	97.02	93.45	3.58	-3.70
3	2.98	6.55	100.00	100.00	-0.00	-3.58

In order to understand the meaning of the clusters we investigate the main positive and negative drivers of features for cluster main customer clusters. The results are shown below.

dataset	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
cluster						
4	32.47	13.95	32.47	13.95	18.52	18.52

Positive

	4
KOMBIALTER	1.06
D19_GESAMT_DATUM	0.66
D19_KONSUMTYP_MAX	0.62
D19_GESAMT_ONLINE_DATUM	0.62

Negative

	4
D19_GESAMT_ANZ_12	-0.54
ONLINE_AFFINITAET	-0.60
D19_VERSAND_ONLINE_QUOTE_12	-0.64
D19_GESAMT_ONLINE_QUOTE_12	-0.68

dataset	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
cluster						
0	12.80	9.95	45.27	23.90	21.36	2.85

Positive

	0
FINANZ_VORSORGER	0.92
CJT_TYP_4	0.83
CJT_TYP_5	0.82
CJT_TYP_6	0.78

Negative

	0
PRAEGENDE_JUGENDJAHRE	-0.88
CJT_TYP_1	-0.93
FINANZ_SPARER	-0.97
FINANZ_ANLEGER	-1.05

dataset	customers_pct	population_pct	customers_pctcum	population_pctcum	diff_cumsum	diff
cluster						
6	11.20	11.21	56.46	35.11	21.35	-0.01

Positive

	6
GEMEINDETYP	1.01
KBA13_AUTOQUOTE	0.94
KBA13_ANTG1	0.82
INNENSTADT	0.79

Negative

	6
KBA13_ANTG3	-0.89
ARBEIT	-0.95
ORTSGR_KLS9	-1.00
EWDICHTE	-1.01

Let’s now have a look on the differences of proportion between customers and general population fir the features that the main positive drivers for customers.

The main obviously differences in positive weights for customers are

- Much more FINANZ_VORSORGER (finance provision)
- CJT_TYP6 and CJT_TYP5 and CJT_TYP4 is much higher (not described)
- KOMBIALTER of customers is higher

The main obviously differences in negative weights for customers are

- FIANZ_SPARER, FINANZ_ANLEGER is much higher
- PRAEGENDE_JUGENDJAHRE are dominated by 40er-60er

Supervised Learning

For the supervised learning we follow these steps

1. **Load Data and clean Data**
   by applying the implemented PreProcessor
1. **Impute missing data**
   Note: Scaling is not required. Do Decision Trees need Feature Scaling? | by Praveen Thenraj | Towards Data Science.

2. **Split Data into Test and validation Data**
   It's important to split the data before we impute missing values in order to avoid that the training data learns from validation data
3. **Train first model**
4. **Compare the segmentation of the datasets**
   We compare the mailout training and test data to the customer data in order to see if the data is clustered similar -> this is just for investigation and not needed for the model

5. **Run GridSearch**
6. Split the given training dataset mailout_train into X_train, y_train where X_train are all features and y_train are the label (RESPONSE column).
7. Build a model pipeline
8. Run GridSearch on the given datasets for different parameters and algorithms

First Supervised Model

We start to train **XGBClassifier** with no tuning on the mailout training data set. In order to validate the result we split the mailout training data into training (80%) and validation data (20%) by random.

The result is a ROCAUC score of 0.5 which is the worst we can get and this confusion matrix

	Pred_False	Pred_True
Act_False	8487	0
Act_True	106	0

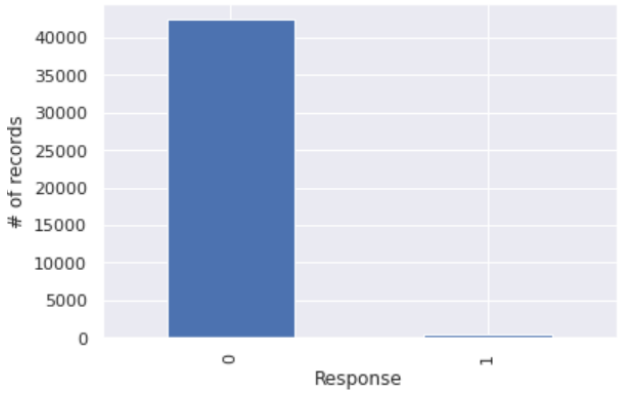
For **RandomForrestClassifier and SVC** the result looks identical.

The **GaussianNBClassifier** it looks a bit better:

```
Classifier: GaussianNB()
ROCAUC score: 0.5633532750421845
```

	Pred_False	Pred_True
Act_False	4278	4209
Act_True	40	66

This is because of a very imbalanced distribution of the labels



Just 426 of 34369 labes are positive which is ~1.2% is labeld with RESPONSE=1.

Because of this we use the BalanceRandomForest classifier. This results in a much better prediction. The ROCAUC score is in this case 0.657 and confusion matrix is

	Pred_False	Pred_True
Act_False	5864	2623
Act_True	40	66

Refinement 1 –GridSearch

In order to improve the results of default parameters of the classifiers we use GridsSearch to train the algorithms for the given training data with different hyperparameters to investigate if this will improve the prediction further.

We run the GridSearch for

- BalancedRandomForestClassifier
    - 'n_estimators': [30,90,250],
    - 'max_depth': [1, 2, 3],
- XGBClassifier
    - 'n_estimators': [30,90,250],
    - 'max_depth' : [1,2,3],
    - ‘learning_rate' : [0.15,0.3],
    - 'scale_pos_weight' : [scale_pos_weight, 1.5*scale_pos_weight],

Scale_pos_weight is the ratio of

(training data with label = 1) / (training data with label = 0)

This search results in a best estimator with the following parameters

```
XGBClassifier(
    learning_rate=0.3,
    max_depth=1,
    n_estimators=30,
    n_jobs=-1,
    scale_pos_weight=80.67840375586854)
```

Using this estimator we receive a ROCAUC score of 0.721 and get this confusion matrix

	Pred_False	Pred_True
Act_False	4956	3531
Act_True	15	91

Compare Segmentation

First let’s check how the segmentation of the training and test data looks compared to the overall customer and general population datasets.

You can see that the proportion of the training data is also higher in the main customer clusters and that the proportion of individuals that have a positive RESPONSE is also higher the main customer clusters 4 and 0 compared to the overall mailout_training dataset. Note that in cluster 4 and 0 the positive RESPONSE quote is much higher compared to the overall mailout distribution share.

Second Model – Using General Population and Customer data as training set

As improvement of the results of the previous model we now use the general population and customer data sets and label them with 0 for general population and 1 for customers. The advantage of this generated dataset is that these data set contains much more data and in particular much more data with positive labels ~150K. We can then test if models trained on this created dataset will work for predictions on the mailout dataset by using the complete mailout training data set as validation set.

Let's start and try this for some classifiers with default parameters.

Using **XGBoost** with default parameters results in a quite good result of

ROCAUC: 0.718

	Pred_False	Pred_True
Truth_False	24419	18011
Truth_True	74	458

Using **RandomForest** is even better with

```
Classifier: RandomForestClassifier()
ROCAUC score: 0.7512887657512861
```

	Pred_False	Pred_True
Act_False	33846	8584
Act_True	157	375

Refinement - GridSearch

Like we did above for the mailout training dataset we will try GridSearchCV again. For the RandomForest this breaks in most cases because of too less memory. Therefore we start to run the GridSearch on the XGBClassifier only.

```
Classifier: Pipeline(steps=[('clf',
                 XGBClassifier(base_score=0.5, booster='gbtree',
                               colsample_bylevel=1, colsample_bynode=1,
                               colsample_bytree=1, gamma=0, gpu_id=-1,
                               importance_type='gain',
                               interaction_constraints='', learning_rate=0.3,
                               max_delta_step=0, max_depth=3,
                               min_child_weight=1, missing=nan,
                               monotone_constraints='()', n_estimators=150,
                               n_jobs=-1, num_parallel_tree=1, random_state=0,
                               reg_alpha=0, reg_lambda=1, scale_pos_weight=1,
                               subsample=1, tree_method='exact',
                               validate_parameters=1, verbosity=None))])
ROCAUC score: 0.7160271052365772
```

	Pred_False	Pred_True
Act_False	24553	17877
Act_True	78	454

We now try a limited GridSearchCV on Randomforest for n_estimators =70 and some different max_depth (1,2,3,None). The result is

```
ROCAUC score: 0.7673042197764031
```

	Pred_False	Pred_True
Act_False	30659	11771
Act_True	100	432

Results

Segmentation

The customer segmentation shows that there is a clear over representation in cluster number four. This cluster contains >32.5% of all customers. Cluster 4,0,6,9 contain more than 67% of all customers. So focusing on these clusters for campaigns and mainly on cluster3 should be result in better response ratio.

Prediction

Using Machine Learning to predict responses we find that the training models on the mailout training dataset will not work for many models with default parameters because of the heavily imbalanced data. Once we are suing classifiers or parameters that consider the imbalance like BalancedRandomForestClassifier or XGBClassifier with scale_pos_weight parameter we get better results.

Using the general population and customer data set to build a new training dataset and validate against the mailout gives much better results. For the RandomForestClassifier with default Parameter we already get a ROCAUC score and confusion matrix like below

```
Classifier: RandomForestClassifier()
ROCAUC score: 0.7512887657512861
```

	Pred_False	Pred_True
Act_False	33846	8584
Act_True	157	375

Using GridSearch for RandomForest for the large dataset failed but worked for very limited search parameter space. Using default parameters on the mailout test result in a Kaggle score of 0.86738 and using

```
{'clf': RandomForestClassifier(n_estimators=70, n_jobs=-1),
 'clf__max_depth': None,
 'clf__n_estimators': 70,
 'clf__n_jobs': -1}
```

gives

```
ROCAUC score: 0.7673042197764031
```

	Pred_False	Pred_True
Act_False	30659	11771
Act_True	100	432

Gives a kaggle score of 0.87088

Improvement

In order to improve the data even more we could try to combine the XGBClassifier, the RandomForestClassifier and the KMeans classification. E.g. the XGBClassifier give this validation result for using the mailout training as validation

	Pred_False	Pred_True
Truth_False	24419	18011
Truth_True	74	458

Whereas RandomForestClassifier gives

```
Classifier: RandomForestClassifier()
ROCAUC score: 0.7512887657512861
```

	Pred_False	Pred_True
Act_False	33846	8584
Act_True	157	375

As you can see XGB has a better sensitivity

Sensitivity = 458 / (74 + 458) = 0.861 (XGBC) compared to 0.705 (RFC)

and RandomForestClassifier a better

Specificity = 33846 / (33646/8584) = 0.798 (RFC) vs 0.576 (XGBC)

References:

Precision Recall curves versus ROC:
https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification/

Metric for imbalanced Data
https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/

Google DS. ROC Introduction
https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc,

Guide for ROC curves
https://towardsdatascience.com/an-understandable-guide-to-roc-curves-and-auc-and-why-and-when-to-use-them-92020bc4c5c1

Clustering

- https://machinelearningmastery.com/clustering-algorithms-with-python/
- https://365datascience.com/tutorials/python-tutorials/pca-k-means/

PCA
https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html