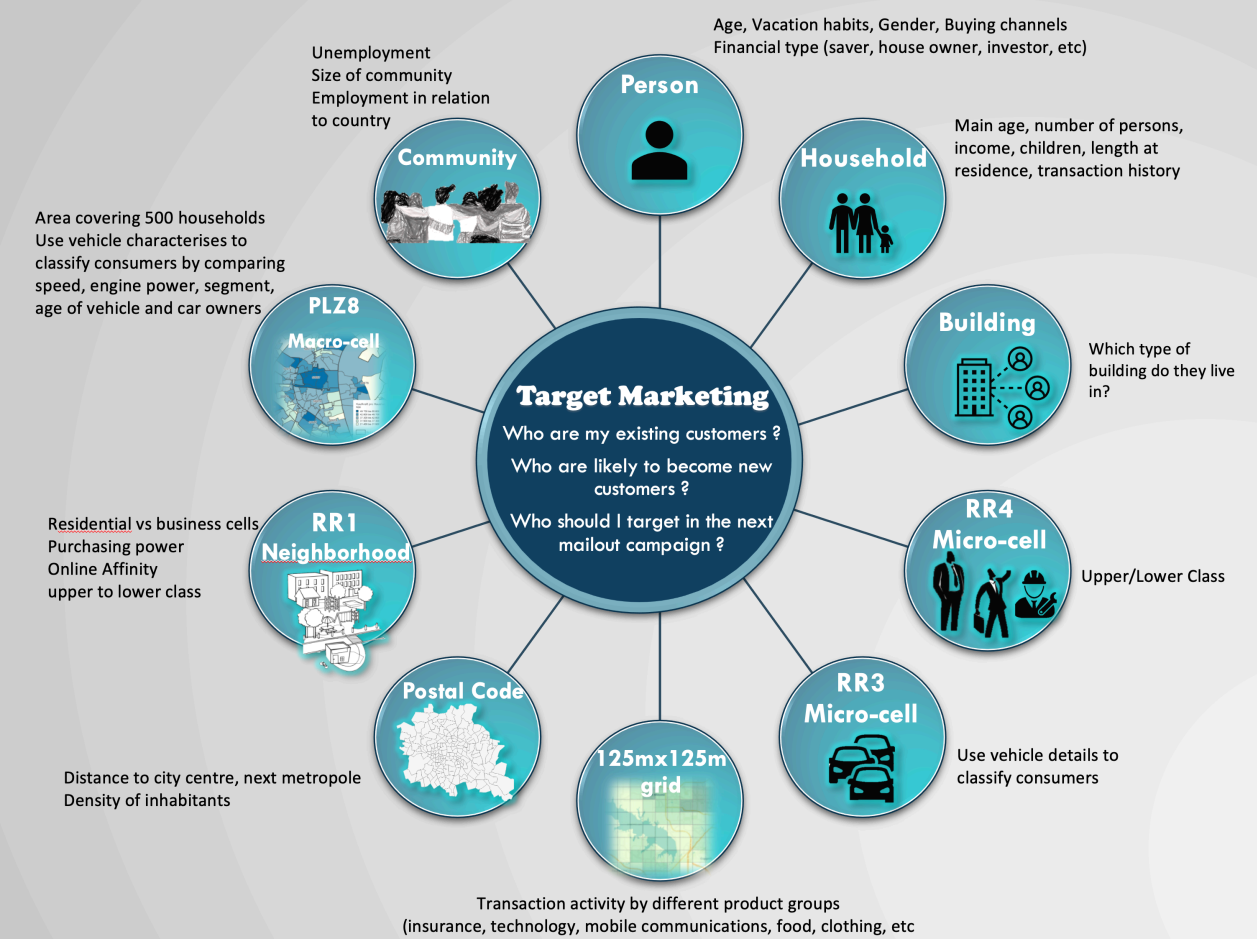
Machine Learning Engineer Nanodegree

Capstone Project Report

Customer segmentation report for Arvato Financial Services



Juanita Smith

14 December 2021

I. Definition

Project Overview

The project selected were sponsored by Udacity in partnership with Arvato Financial Solutions.
The project are within the marketing domain, with main objective to help one of Arvato's clients, a mail-order company in Germany, acquire more clients easier and smarter.
The mail-out company are running mail-out campaigns, with their objective to increase efficiency in their customer acquisition process by targeting the right people.

The was data was provided by Arvato and is protected under Terms and Conditions.
This is a real data-science project with real data.

Problem Statement

From business perspective, Arvato needs help to answer 3 questions for their mail order sales client:

1. Help the mail order company to understand who their customers are
2. How does their client base compare to the rest of Germany ?
3. How can the mail order company acquire new clients more efficiently ?

The project is divided into 3 main parts:

Part 1: Data Exploration and cleaning

Data is coming mainly from open source data, thus lots of cleaning and preparation work is needed before we can start modelling.

Part 2: Customer segmentation

Analyse the demographics data for customers of the mail-order company in Germany, comparing it against demographics information for the general population. Use unsupervised learning techniques to perform customer segmentation, identifying the parts of the population that best describe the core customer base of the company.

Part 3: Mail-out campaign prediction

Apply what was learned on a third dataset, which contain demographics information for individuals who were targeted in a marketing campaign. Build a model to predict whether an individual will respond to the mail-out campaign or not.

Metrics

The following metrics will be applied to each phase of the project to validate model performance

1. Clustering in PART 2: Dimension reduction with PCA - Keep components that explain minimum 85% of the variability of the data
2. Clustering in PART 2: Clustering with KMeans - Use elbow method to select the number of clusters where the average distance of the cluster points to cluster mean does not significantly decrease anymore when we add more clusters.
3. PART 3: AUC (Area under the Curve) for mail-out response prediction

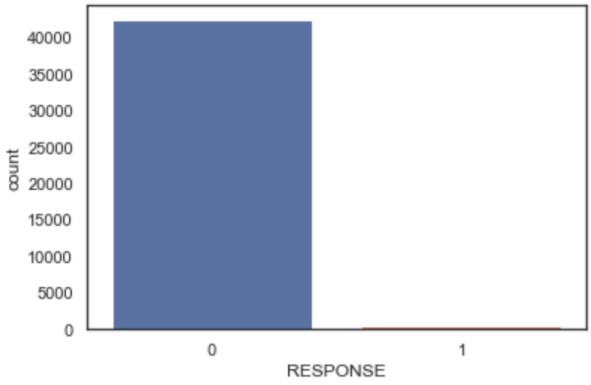


Fig 1.1

There is a large class imbalance in the data, where most individuals did not respond to the mail-out campaign. In the training dataset, we can see only 517 out of 12889 individuals responded the campaign, equating to only 3% !
Predicting individual classes using accuracy when the dataset is imbalanced, is not an appropriate performance evaluation method. Instead, the kaggle competition requires metric AUC (to be used.

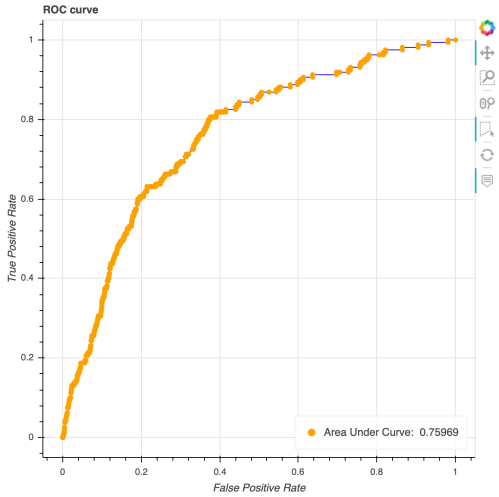


Fig 1.2

AUC graph plots the ratio of positive errors over negative errors. Its goal is reduce the overall amount of errors made. The curve plots the ratio of True Positive rate over the False Positive rate
TRP = True Positives / All Actual Positives
FPR = False Positives / All Actual Negatives

	Predicted **0**	Predicted **1**
Actual **0**	TN *9008*	FP *3721*
Actual **1**	FN *45*	TP *115*

Fig 1.3

Below are an example of a confusing matrix and the formula's to explain what TPR and FPR represent

TPR = TP / TP + FN = 115 / (115 + 45)

FPR = FP / FP + TN = (3721 / (3721 + 9008)

II. Analysis

Data Exploration

6 Datasets were provided by Arvato. 4 Datasets in csv format contain the same demographic data and have similar layouts. They contain over 360 features which contain german heading names. It's very difficult to understand and interpret this data due number of features and language, and therefore the last 2 excel files provided which describe the data and each feature in detail are key to the project. These description files were integrated into data exploration to understand and interpret the data in an automated way.
Below a brief description of each file with basic visualisations

1. Feature Levels / Categories

Each feature is mapped to a category/level, describing different aspects of the demographic data from looking at the individual, his/her immediate household, financial status, buildings/houses they live in, and the different cells around the area they live in to understand the immediate neighbourhood but also the wider area. (See visualisation on the title page for a full description of the levels).
Example of the data in fig 2.1

Attribute	Information level	Description	Additional notes
AGER_TYP	NaN	best-ager typology	in cooperation with Kantar TNS; the information basis is a consumer survey
ALTERSKATEGORIE_GROB	Person	age through prename analysis	modelled on millions of first name-age-reference data
ANREDE_KZ	Person	gender	modelled on millions of first name-age-reference data
CJT_GESAMTTYP	Person	Customer-Journey-Typology relating to the preferred information and buying channels of consumers	relating to the preferred information, marketing and buying channels of consumers as well as their cross-channel usage. The information basis is a survey on the consumer channel preferences combined via a statistical modell with AZ DIAS data
FINANZ_MINIMALIST	Person	financial typology: low financial interest	Gfk-Typology based on a representative household panel combined via a statistical modell with AZ DIAS data

Fig 2.1

The excel file was manually enhanced to correct column names so it can be matched to other files and data in next steps. Example of corrections made:
- Remove suffix '_RZ' from 125x125 grid level fields (e.g. D19_TECHNIK_RZ to D19_TECHNIK)
- correct spelling of field name D19_BUCH_CD (e.g. D19_BUCK_RZ to D19_BUCH_CD)

A lot of features were not mapped to levels and had to be derived using google translations

2. Feature descriptions

This dataset contains a detailed description for values within a feature. As the dataset contains mostly ordinal ranges for example low too high, this dataset is key to understand the data.

Example of the data:

Attribute	Description	Value	Meaning
MOBI_REGIO	moving patterns	1	very high mobility
MOBI_REGIO	moving patterns	2	high mobility
MOBI_REGIO	moving patterns	3	middle mobility
MOBI_REGIO	moving patterns	4	low mobility
MOBI_REGIO	moving patterns	5	very low mobility
MOBI_REGIO	moving patterns	6	unknown

Fig 2.2

File was manually enhanced in Excel to identify additional missing values where it's needed example:

- MIN_GEBAEUDEJAHR = 0 (year the building was first mentioned in our database)
- ALTER_HH = 0 (Age)
- D19_LETZTER_KAUF_BRANCHE = 'D19_UNBEKANNT'

3. Udacity_AZDIAS_052018.csv

Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns).
Column = 'LNR' is a unique key for each individual with value range min 191653 to max 1082873

4. Udacity_CUSTOMERS_052018.CSV

Demographics data for customers of the mail-order company; 191 652 persons (rows) x 369 features (columns).
This dataset is looking the same as AZDIAS dataset above, except for 3 additional columns 'CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP' with was dropped for unsupervised learning.

Column = 'LNR' is a unique key for each individual with value range min 1 to max 191652. We thus conclude completely different individuals appear in the two datasets thus there are no overlaps

5. Udacity_MAILOUT_052018_TRAIN.csv

Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).
Contains the same data as AZDIAS above, except for a 'RESPONSE' column to indicate if the individual responded to the mail-out campaign or not. This dataset will be used to train a prediction model.
Column = 'LNR' is a unique key with values falling into the same range as in the customer dataset.
We thus conclude that all individuals that are in the training dataset are also in the customer dataset.

6. UDACITY_MAIlOUT_052018_TEST.csv

Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).
Dataset contain the same data as AZDIAS and TRAINING dataset above, however the 'RESPONSE' column is omitted. This means we cannot measure final performance by ourselves, we have submit the prediction to Kaggle for evaluation.
Column = 'LNR' is a unique key with values falling into the same range as in the customer dataset.
We thus conclude that all individuals that are in the testing dataset are also in the customer dataset.
Sample layout for datasets 4-6 for first few columns

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALTE_AK
90	21511	3.0	1.0	14.0	NaN	NaN	NaN	NaN	10.0	
129	61905	2.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
173	15467	1.0	1.0	13.0	NaN	NaN	NaN	NaN	13.0	
205	25211	1.0	1.0	9.0	NaN	NaN	NaN	NaN	8.0	
248	83461	1.0	1.0	11.0	NaN	NaN	NaN	NaN	10.0	

There are around 62 fields not described in the features dataset, and the levels for 46 attributes are absent. These values are not dropped by default as they might be important. Google translate was used to infer the values were possible.

Exploratory Visualisations

The data provided comes mostly from open source data, which are extremely messy, with lots of missing data. A lot of effort was needed to prepare the data for modelling.

There are 3 types of missing values:

1. Where data is *truly absent* in the dataset - identified with *np.nan*
2. Where data is marked with label 'unknown' in the features excel file
    - During data loading, we immediately convert such values to np.nan and we will impute them
3. Where data are marked with labels below in the attributes file
    - 'no transactions known', 'no transaction known', 'no Online-transactions'

When comparing missing data in columns that are over 30% in the different datasets in fig 2.4, we can see that generally, the most missing data are present in the full population dataset.

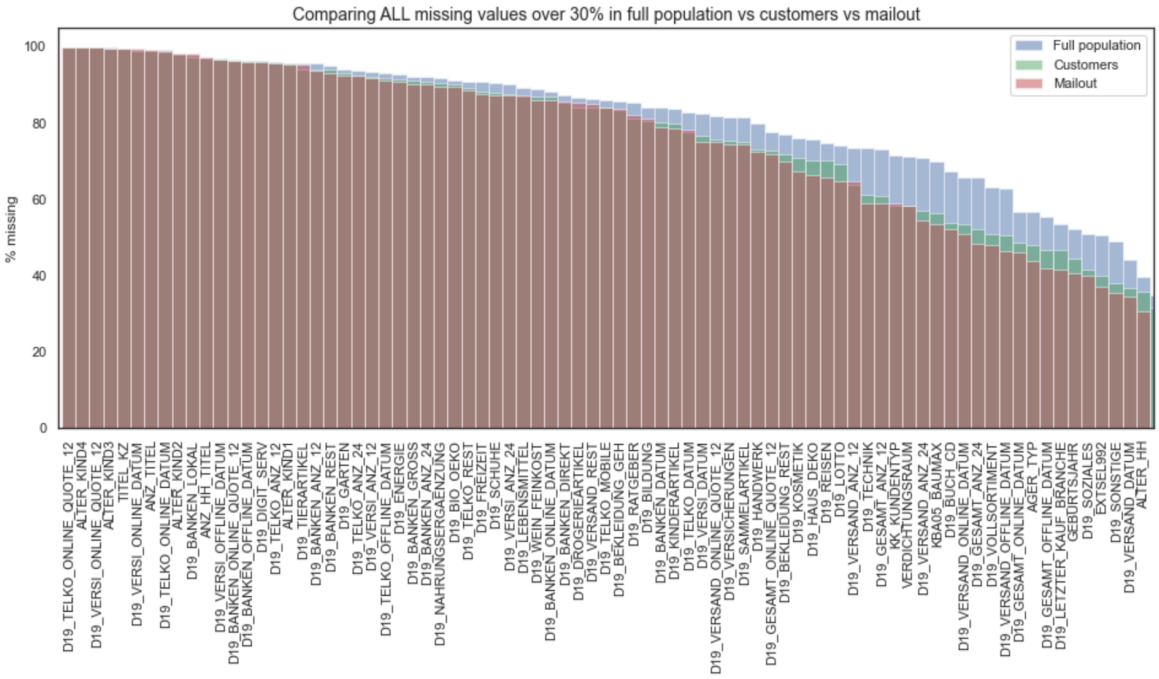


Fig 2.4

It can be observed columns starting with 'D19_* which represent grid level data, generally contain very little data (e.g. meaning 0 transactions present for buying clothes). We can also observe, the mail-out and customer dataset contains a lot less missing values.

One cannot just look at missing values in the general population dataset in isolation because this is a hugely imbalance dataset (Meaning only 517 out of 12889 individuals responded). We need to balance missing data in a feature together with how many people responded to mail-out campaign to make sure we don't just throw valuable information away.

To Illustrate, lets look at feature ‘D19_SOZIALES’
In Fig 2.5.1, we observe the imbalance of 0 transactions vs other values. On first glance we would be tempted to drop the column.

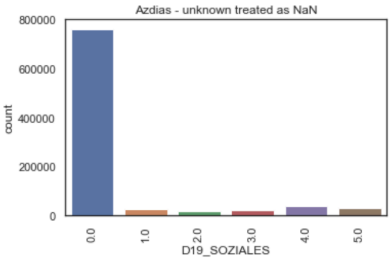


Fig 2.5.1

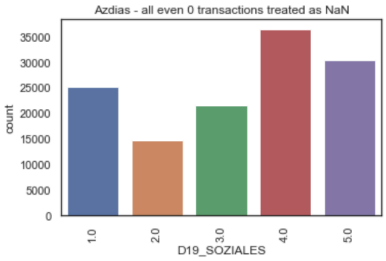


Fig 2.5.2

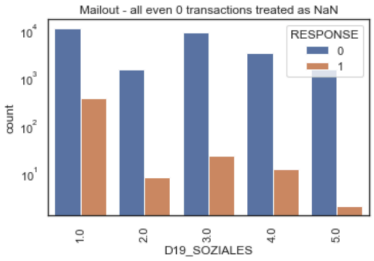


Fig 2.5.3

In Fig 2.5.2, if we treat ‘0’ as null value we can more easily observe the distribution of data over the other values. In Fig 2.5.3, we observe those who have responded to the mailout campaign. Out of 517 of those individuals who have responded, 375 of D19_SOZIALES = 1 responded. We definitely should not drop this column.

A new dataset was build to bring together a full picture of each feature showing its level, descriptions in English, missing values (based only all 3 categories described above), distinct values and it’s ranges.
This helped a lot to build a general understanding on this monster dataset.

Example extract:

Attribute	distinct_values	unique_values	Information level	Description	Additional notes	feature_description	%missing_values
VERDICHTUNGSRAUM	45	[nan, 1.0, 35.0, 3.0, 7.0, 23.0, 4.0, 8.0, 13.0, 16.0, 25.0, 5.0, 21.0, 6.0, 15.0, 32.0, 42.0, 31.0, 11.0, 33.0, 22.0, 30.0, 18.0, 12.0, 27.0, 2.0, 9.0, 28.0, 10.0, 14.0, 20.0, 17.0, 43.0, 19.0, 24.0, 34.0, 40.0, 39.0, 29.0, 26.0, 44.0, 45.0, 37.0, 36.0, 41.0, 38.0]	NaN	NaN	NaN	population density	52.29
CAMEO_DEU_2015	44	[nan, 8A, 4C, 2A, 6B, 8C, 4A, 2D, 1A, 1E, 9D, 5C, 8B, 7A, 5D, 9E, 9B, 1B, 3D, 4E, 4B, 3C, 5A, 7B, 9A, 6D, 6E, 2C, 7C, 9C, 7D, 5E, 1D, 8D, 6C, 6A, 5B, 4D, 3A, 2B, 7E, 3B, 6F, 5F, 1C]	Microcell (RR4_ID)	CAMEO_4.0: specific group	New German CAMEO Typology established together with Call Credit in late 2015	CAMEO classification 2015 - detailled classification	11.15
LP_LEBENSPHASE_FEIN	40	[15.0, 21.0, 3.0, nan, 32.0, 8.0, 2.0, 5.0, 10.0, 4.0, 6.0, 23.0, 12.0, 20.0, 1.0, 11.0, 25.0, 13.0, 7.0, 18.0, 31.0, 19.0, 38.0, 35.0, 30.0, 22.0, 14.0, 33.0, 29.0, 24.0, 28.0, 37.0, 26.0, 39.0, 27.0, 36.0, 9.0, 34.0, 40.0, 16.0, 17.0]	Person	lifestage fine	modelled on different AZ DIAS data	lifestage fine	10.95
EINGEZOGENAM_HH_JAHR	37	[nan, 2004.0, 2000.0, 1998.0, 1994.0, 2005.0, 2007.0, 2009.0, 2016.0, 2014.0, 2015.0, 2013.0, 2008.0, 2010.0, 2001.0, 2002.0, 1997.0, 2012.0, 1992.0, 1999.0, 1996.0, 1995.0, 2011.0, 2003.0, 2006.0, 1991.0, 2017.0, 1993.0, 2018.0, 1989.0, 1990.0, 1987.0, 1986.0, 1988.0, 1900.0, 1904.0, 1971.0, 1984.0]	NaN	NaN	NaN	None	8.25

Fig 2.6

Missing values by level reveal that grid and household columns have lots of 0 transactional data which is a watch out.

Information level	
125m x 125m Grid	85.628485
Building	10.332500
Community	10.916667
Household	63.366286
Microcell (RR3_ID)	17.268333
Microcell (RR4_ID)	15.012727
PLZ8	11.945089
Person	6.037805
Postcode	10.520000
RR1_ID	12.286000

Missing data in ROWS in the different datasets revealed a similar pattern, some rows contain over 60% of missing data which where dropped but only in the full population dataset as this is the base to build and train the models from.

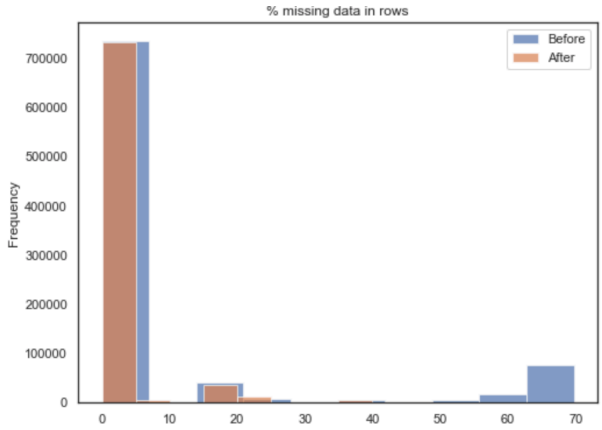


Fig 2.7

Algorithms and Techniques

To solve the problem the following steps will be followed:

1. As the datasets have over 360 features PCA (Principle Component Analysis) will be used to reduce the dimensionally of the dataset. A scree plot will be used to analyse how each PCA component contribute to the total variance explained. Aim to keep components that explain total variance of around 85-90% of the variance.
2. Train a model to cluster the general population of Germany using ski-kit learn's KMeans class to perform clustering on the PCA-transformed data. Use the elbow method to select the most optimal number of clusters.
3. Use the trained KMeans model to cluster the customers of the mail order company. By comparing the clusters we should be able to describe the relationship between the demographics of the company's existing customers and the general population of Germany. By the end of this part, it should be possible to describe parts of the general population that are more likely to be part of the mail-order company's main customer base, and which parts of the general population are less so.
4. In the last step, use the clusters each customer belongs to, in additional with demographic data to train a supervised model to predict which individuals will respond to mail-out campaign or not. Tree based algorithms will be selected as we are dealing with a binary logistics problem and a dataset containing mostly categorical features. First, scikit-learn's tree-based algorithms DecisionTree, AdaBoost and XGBClassier will be used as quick comparison to set a baseline. XGBoost is the main algorithm chosen to refine results. XGBoost involves creating and adding trees to the model sequentially. New trees are created to correct the residual errors in the predictions from the existing sequence of trees. The effect is that the model can quickly fit, then overfit. One key strength of the algorithm is its ability to apply regularisation on the features, meaning it could get rid of useless features or shrink importance of features that cause bias. As the datasets have over 400 features after one-hot encoding, this becomes extremely important. XGBoost is a competition winning algorithm on Kaggle, I have high expectations !

Benchmark

Training dataset was split into a training and validation dataset in 70/30% ratio, using a stratified split to handle the imbalance, making sure those candidates who responded are represented proportionally in both training and validation datasets.

As benchmark, scikit-learn's algorithms have been chosen which solve binary classification problem. Decision trees are easy to understand, train fast, but can easily overtrain. It fits well to the large dataset full of mostly categorical and ordinal features.
DecisionTreeClassifier, AdaBoostClassifier and XGBClassier was selected for comparison.
Without any hyper parameter tuning, it can be observed that XGBClassier performed slightly better with **64% AUC on the validation dataset, which will use as a benchmark.**

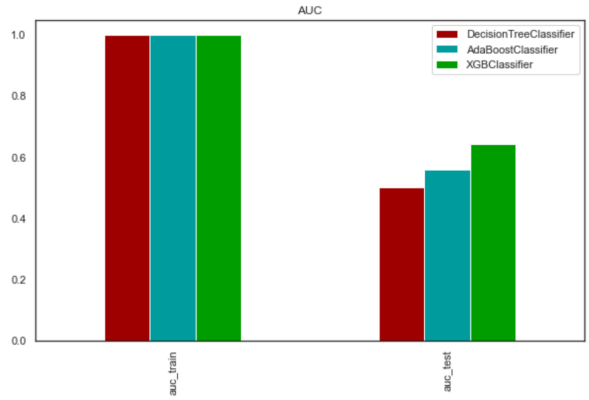


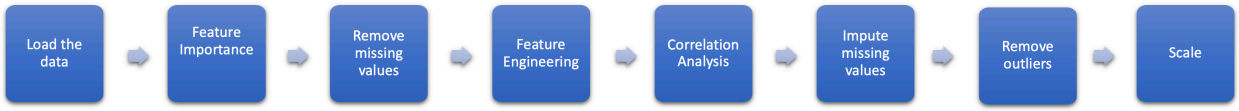
Fig 2.8

In Fig 2.8, we can also observe the AUC on the training dataset is 100% on all 3 algorithms, which indicates that the model is overfitting.

III. Methodology

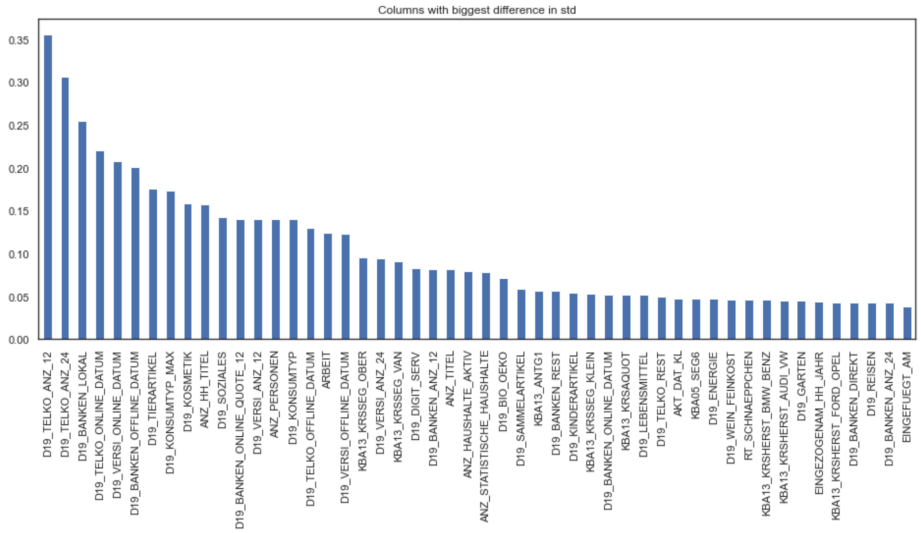
Data Preprocessing

The following sequential steps are followed to clean the data. Decisions made during data explorations are used to build one general cleaning function and to build configuration files of which columns to drop, encode for supervised and unsupervised irrespectively.



2. First look at Feature Importance

We need to take care, as even though missing values might be high, if there are strong distinguishing columns we should avoid to drop them. In below chart we compare the standard deviation in features between those individuals who responded to the market campaign vs not.



Notice that D19_KONSUMTYP_MAX and D19_SOZIALES are in the top 10. They are the perfect example why we should take care, as they also contain lots of 0 transactional data.

The top 80 columns from this analysis will not be dropped. However time was invested in sophisticated imputing methods below.

Fig 3.1

Remove missing values

Drop columns with missing values exceeding 50% in the full population dataset .
Drop columns where ALL missing values (also transactions = 0) exceed 80% in the mailout success dataset.

Feature Engineering

There are a handful of features that are "mixed" that require special treatment in order to be included in the analysis.

Some examples below

- "PRAEGENDE_JUGENDJAHRE" combines information on three dimensions: generation by decade, movement (mainstream vs. avantgarde), and nation (east vs. west). While there aren't enough levels to disentangle east from west, 2 new variables are created to capture the other two dimensions: an interval-type variable for decade, and a binary variable for movement.

- "CAMEO_INTL_2015" combines information on two axes: wealth and life stage. Break up the two-digit codes by their 'tens'-place and 'ones'-place digits into two new ordinal variables.
- Feature 'WOHNLAGE' combines information about residential and rural area's. Break up the feature into 1 ordinal variable describing residential neighbourhood quality from high to low, and a binary variable indicating if this is a rural area or not
- Strip feature 'PLZ8-BAUMAX' and 'KBA05_BAUMAX' into an ordinal feature describing only the number of family houses in an area, and introduce a new binary feature indicating if it is a business building or not

Configuration files were created with columns to one-hot-encode as the values did not represent an ordinal pattern from low to high, or high to low

Additional ordinal columns were one-hot encoded to help interpret the meaning of clusters and predictions more easily. It does not impact the clustering results, but brings better explainability to describe the cluster to business audience.

Correlation Analysis

Correlation analysis was done after feature engineering in order to catch relationships between new features and existing ones.

Determine which features are too highly-correlated with each other to include both features in a single model. Correlations with score > 90 were removed.

Example: Family, Lifestyle and Status

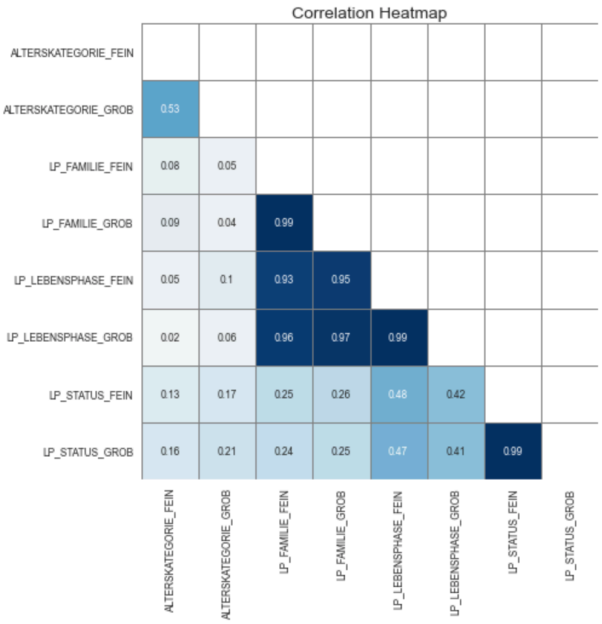


Fig 3.2

Keep the fields that split out age, status and income clearly and separately meaning:
·ALTERSKATEGORIE_GROB (< 30 years, 30-45, 46-60, >60)
·LP_FAMILIE_GROB (single, couple, single parent, family, multiplerson household)
·LP_FAMILIE_FEIN (same as GROB but describe age of children young, teenage or full age.
·LP_STATUS_GROB (low-income earners, average earners, independents, houseowners, top earners)

Impute missing values

IterativeImputer was used to impute missing values, a machine learning approach which estimates each feature from all the others. Each feature is modelled as a function of the other features, e.g. a regression problem where missing values are predicted. Each feature is imputed sequentially, one after the other, allowing prior imputed values to be used as part of a model in predicting subsequent features.

15 maximum rounds of training were selected, and 50 nearest features due to amount of features and missing values. I used mean as initial strategy for numerical fields and most_frequent as initial strategy for all other fields.

When using IterativeImputer, the distribution curve before and after after are similar and missing values are spread over the curve. See example distribution plots below comparing original data in the first plot in Fig 3.3.1, SimpleImputer with most_frequent in the middle in Fig 3.3.2 and IterativeImputer as the 3rd plot in Fig 3.3.3. It illustrate the benefits pretty well. SimpleImputer are introductions peaks and possible bias.

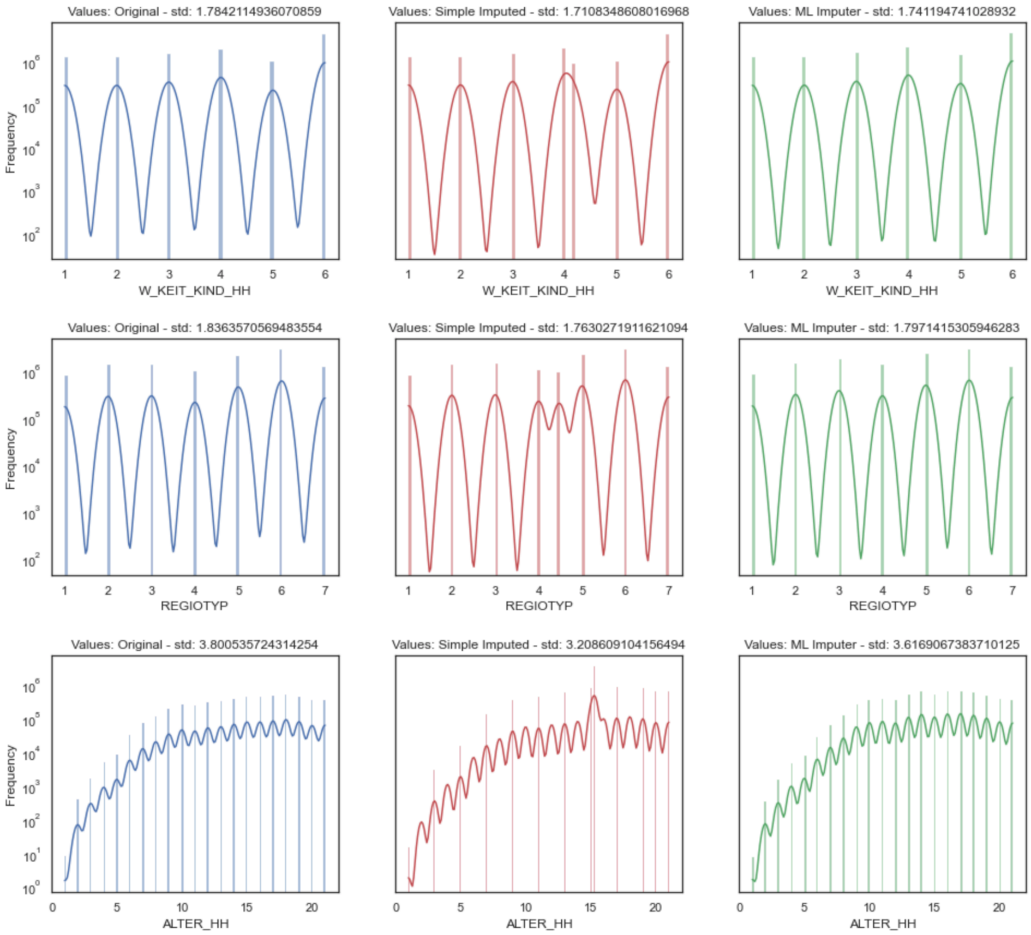


Fig 3.3.1 *Fig 3.3.2* *Fig 3.3.3*

Removing Outliers

Some outliers like EINGEZOEGNAM_HH_JAHR was detected with only few data points prior year 1995. These outliers was merged with 1995 data now distribution looks a lot better

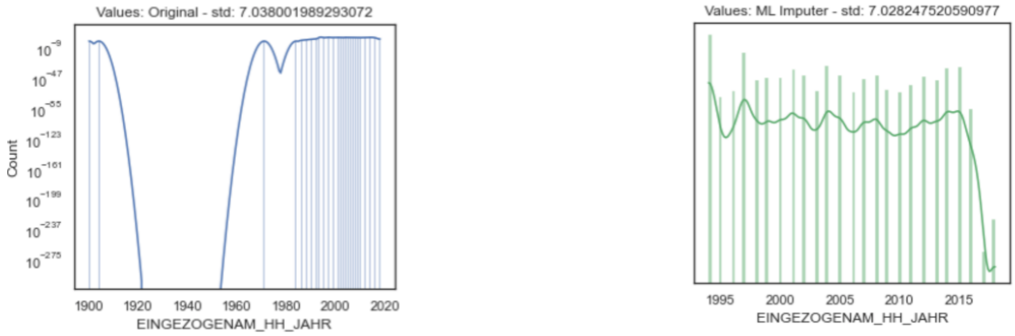


Fig 3.4.1 *Fig 3.4.2*

Implementation - Unsupervised learning

Clustering

PCA (Principal Component Analysis) was used to reduce the dimensionality of the data. PCA's goal is not feature reduction, instead we try to summarise features to build new features which we call 'principle components'

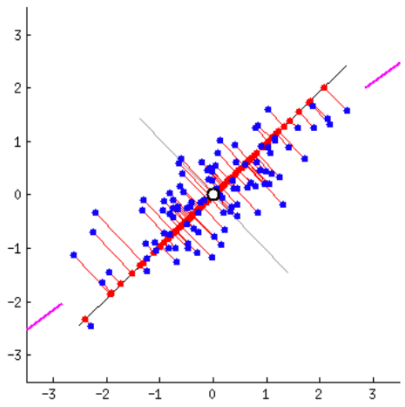


Fig 3.6

PCA is just another way to summarise data.

Within each PCA component we are looking for features with strong linear relationships yet has the highest amount of variation that describes the differences across the population the best.

PCA components have linear relationships inside the component, but have no linear relations between each other, they are orthogonal, means they are independent from one another.

Within the scope this project, we can think of each component as a *type of persona* describing certain characteristics of individuals in terms of mainly where they live, financial and family status, age and interests

In below graph, when we display all 400+ components we can see how much of the variance can be explained by each component. We observe the first component describe 8.3%, and then there is a steep drop to around 2.5%. Around 200 components very little variance can be observed.

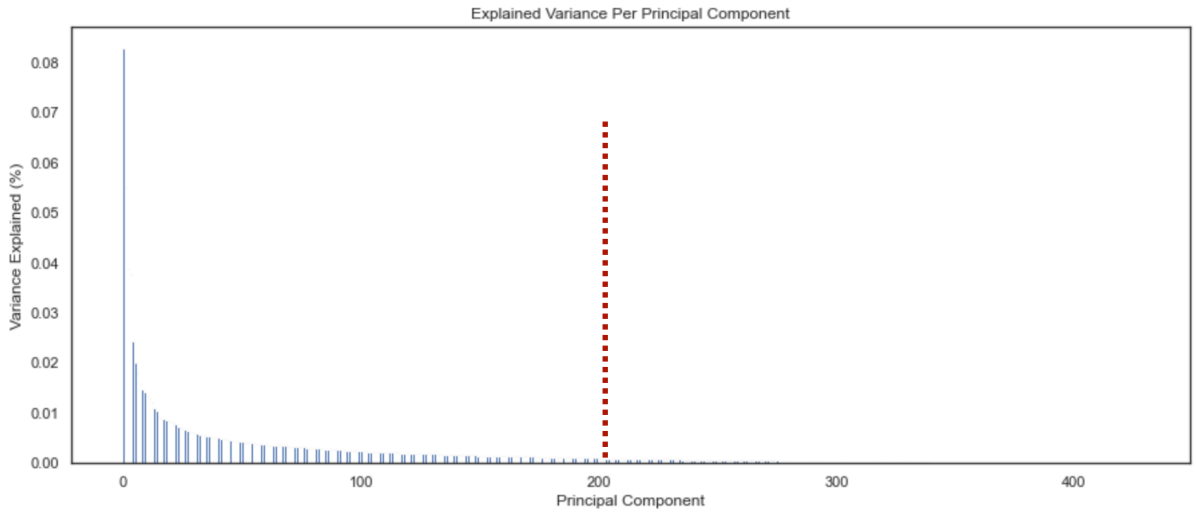


Fig 3.7

The goal was to find the number of components that explain around 90% of the variance of the data. which was represented by 180 components.

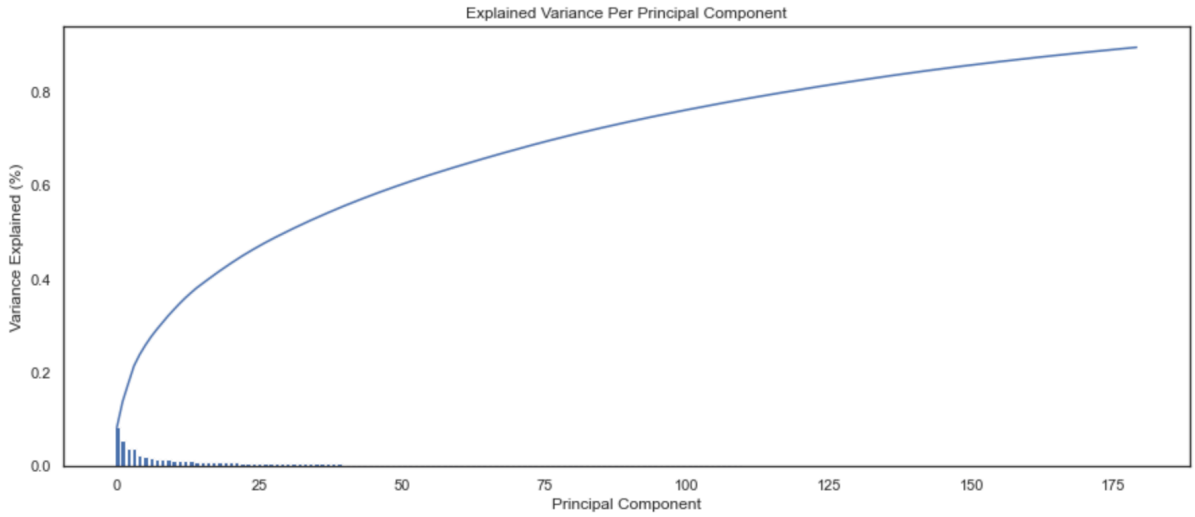
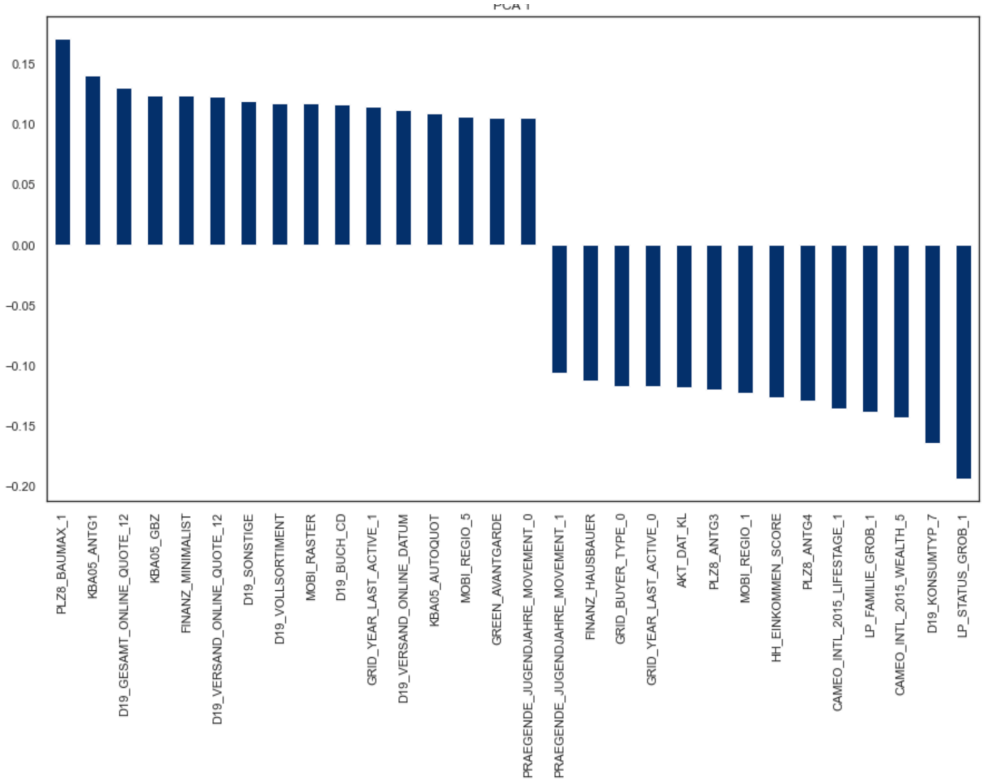


Fig 3.8

Analysing the first 5 components, which each describe a kind of persona:

PCA component 1 uses mainly building type and geographical location, mobility and financial status to describe this persona and explain 8.1% of the variance when we select 180 components

Analysing the strongest positive and negative linear relationships this component is describing top earner families living in area's with mostly 1-2 family houses which they own, outside of the city but in larger residential area's. They own multiple cars per household. They are active online customers, yet minimalistic, and are part of the green avantgarde.



PCA component 2 is focussing on youth movement and age, financial status and online shopping behaviour and explain 5.6% of the variance when we use 180 components

This component describe digital media kids, adults who's youth was in their 90's. Younger low income families with high probability of having children. They have a super strong online presence, seems to shop online frequently and react well to advertising campaigns and catalogues. They don't save or invest, but in contrast wants to be financially prepared.

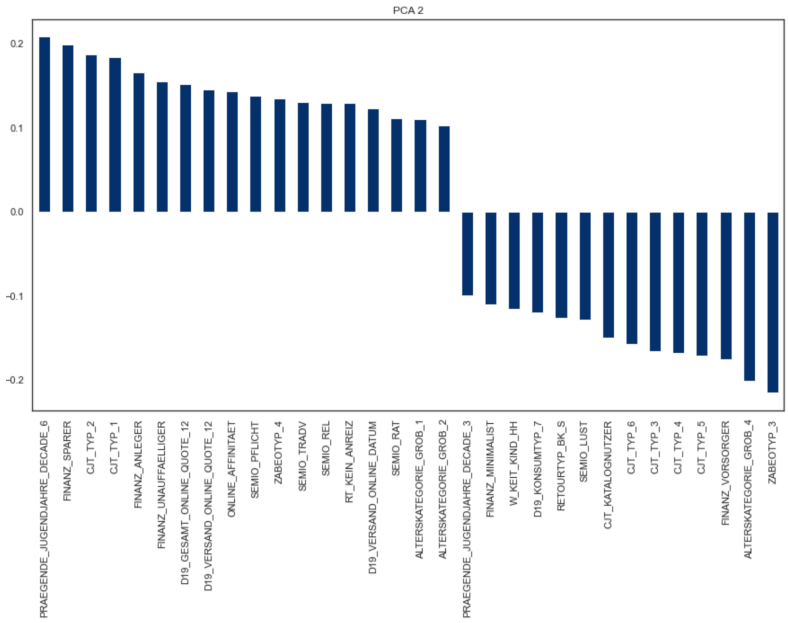


Fig 3.9

PCA component 3 persona describes individuals that are living in dense populated area's in West-Germany near the city centre. They live in area's where there are high number of > 6 family houses but also lots of businesses present. They drive top and middle class German manufacturer car brands mostly smaller BMW and Mercedes with <= 4 seats.They are part of the green avantgarde. These individuals are not in age group 46-60, and are most likely to be pre-family and single younger people. They were active online in the last year. The component explain 3.9% of the variance.

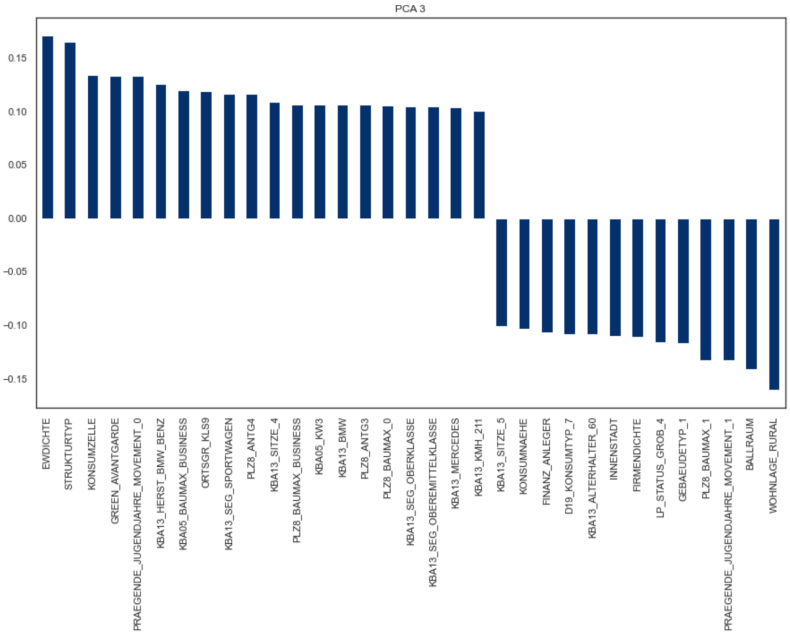


Fig 3.10

PCA component 4 persona is describing older people > 60 years living in East-Germany from poorer households and lower income and not part of green avantgarde. They are driving smaller mid-class vehicles like Ford, Masda and of Asien origin with 5 seats. They are frequent online shoppers buying insurance, technology and respond to complete mailout offers and were active in the last year. They live in area's where there are a high share of 6-10 family houses in the area.

PCA Component 5 is using mainly personal interest to describe the persona.
This persona represent males, fight-full, dominant, critical and rational, but not dreamy, cultural, family orientated, social or religious. They hate shopping in general and are frequent users of advertising catalogues. They are unlikely to be 46-60 years old. Online shopping are a great convenience for those who hate going to the shops

180 PCA components was used to cluster the data using KMeans. The elbow method was used to find the most optimal number of clusters, where the average distance from each point to its cluster centre no longer significantly decrease. From the figure below we can see there is no clear elbow. There seem to be steep fall up to cluster 5, then the curve slowly levels off. Around cluster 7-8 the steepest curve seems to start levelling off. 8 Clusters was selected.

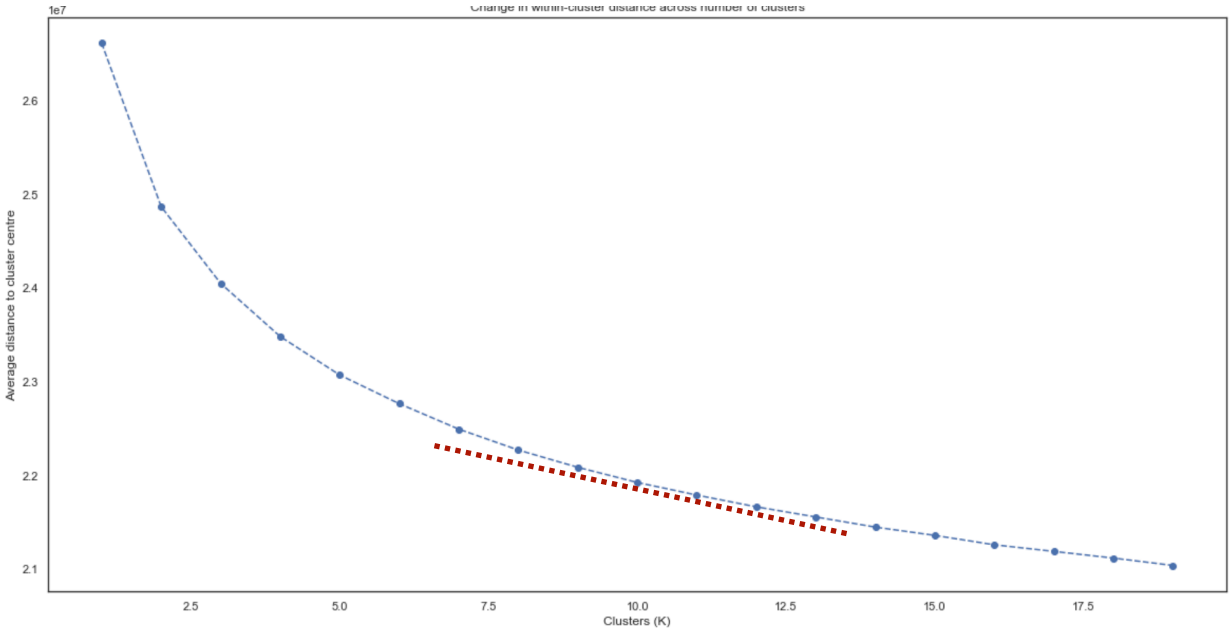


Fig 3.11

When plotting the first 3 PCA components coloured by assigned cluster for both the full population and the customer base of the mail order company, we can observe the points (which each represent an individual) are well separated and the clusters are clearly defined.

We can also visually observe that INDIGO (cluster 0) is the most over represented in the customer base, where PINK (cluster 7) is the most under represented in the customer base and hardly visible

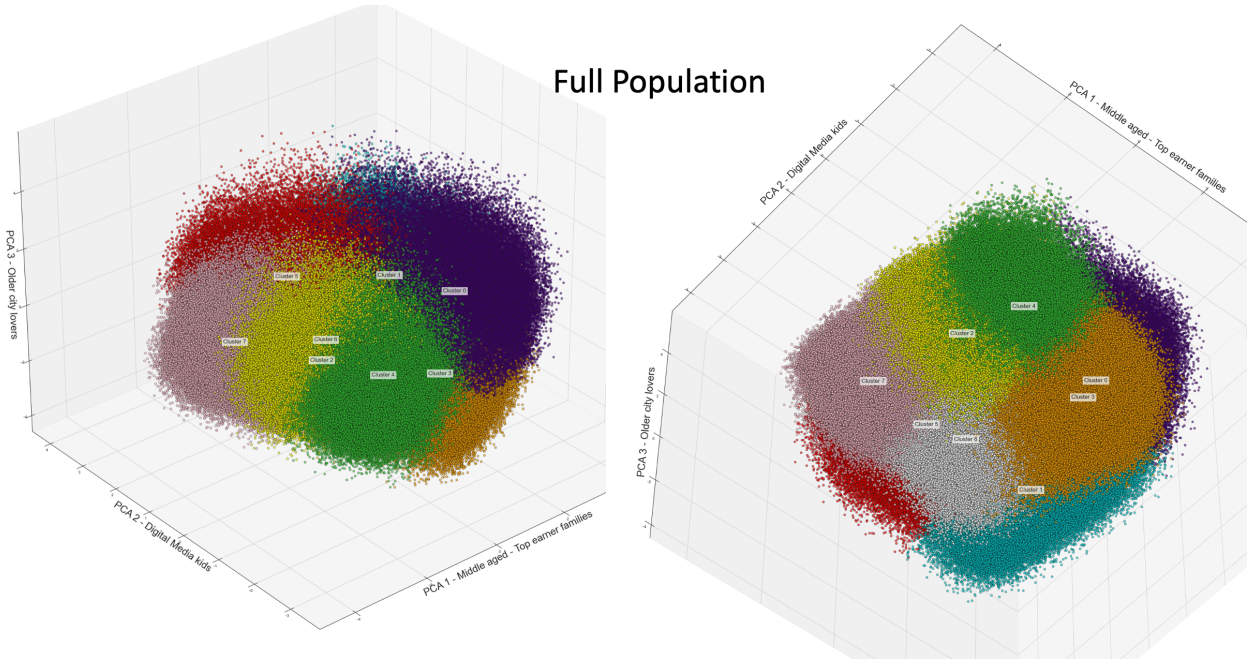


Fig 3.12

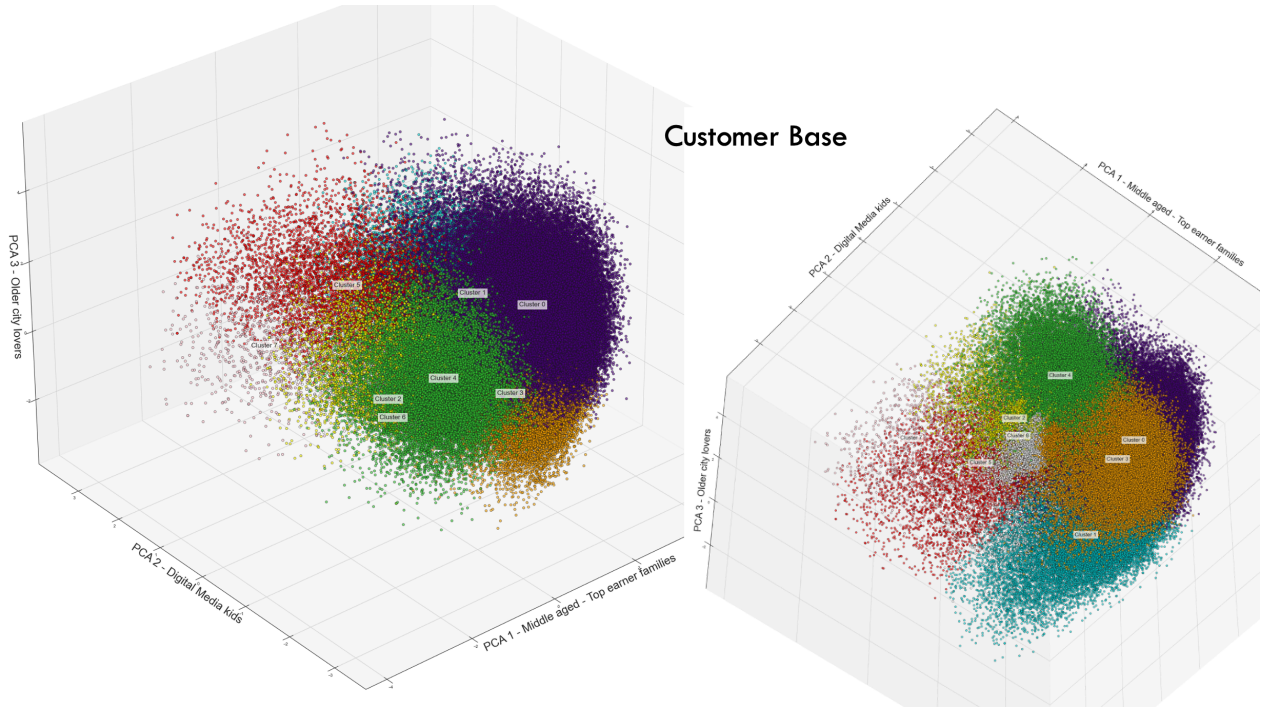


Fig 3.13

Plotting the clusters we can observe the clusters in the full population are quite evenly distributed, where in the customer base the we have a lot of variation.

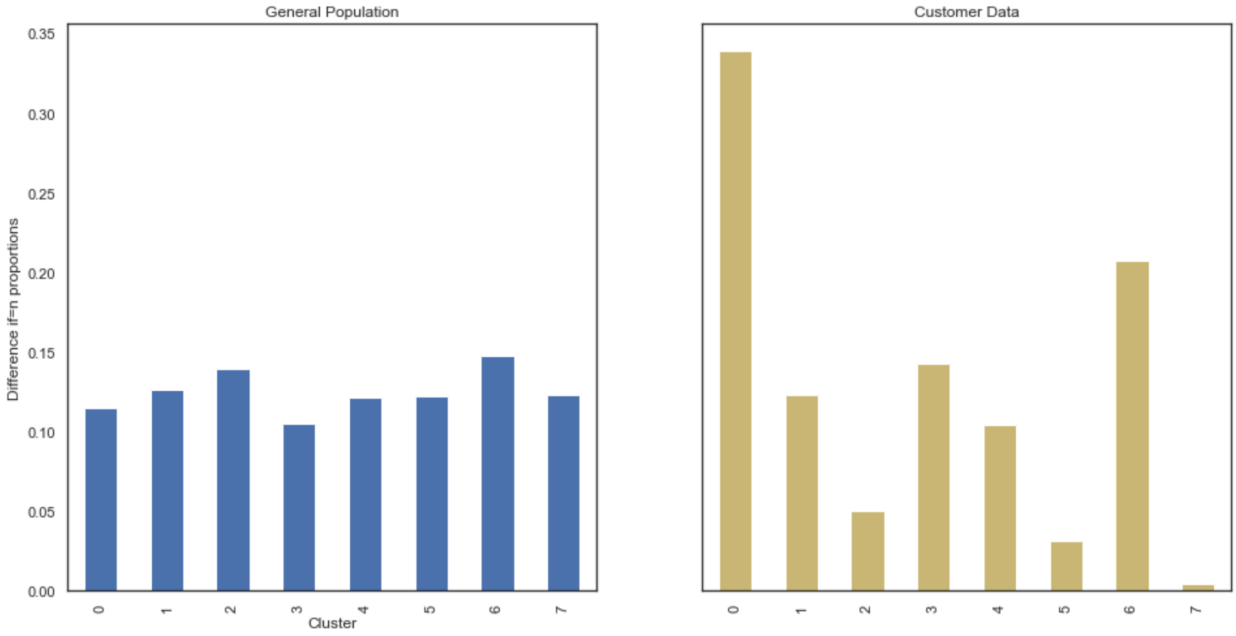


Fig 3.14

We can observe cluster 0 are the most over represented, and cluster 7 are the most under represented, which correspond to the drawings on the previous page (INDIGO and PINK)

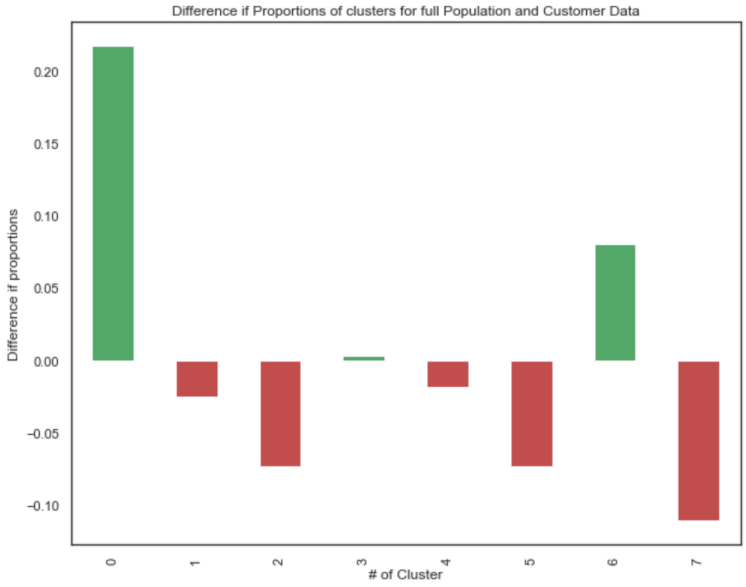


Fig 3.15

Analysis cluster 0 (most over represented in customer base)
PCA component 1 and 3 contain the biggest positive weight in cluster 0, whilst component 2 and 4 have the largest negative weight.

This suggest that cluster 0 are focussing on the high-income earners, which are either settled top earning families living outside of the city, or individuals from West-Germany driving top class german manufacture vehicles living near or in the city centre.

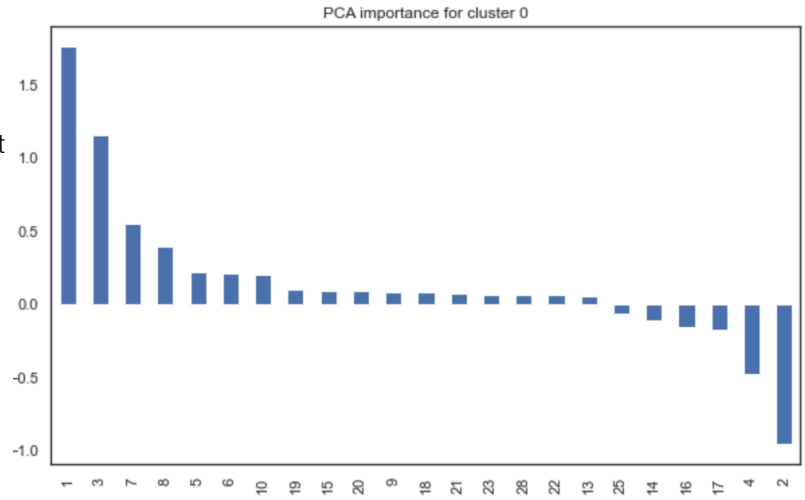


Fig 3.16

Analysis of cluster 7 (most under represented in customer base)
Cluster 7 have a very strong negative weight for PCA component 1 which indicates that top earners families are totally excluded.

Largest positive weight is PCA component 2, which represent young low income families. Also component 5 is featuring means those dominant males who hates going to the shops and prefer online shopping, are included in the customer base of the mail order company.

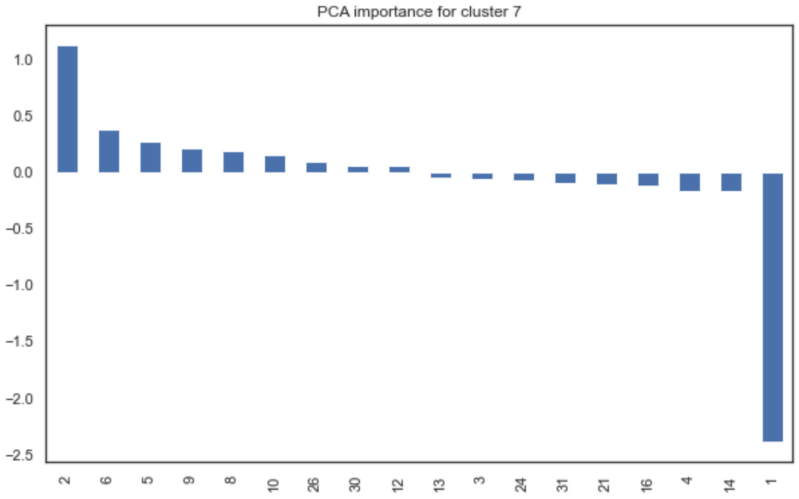


Fig 3.17

Implementation - Supervised learning

When combining the clustering result with the mail-out campaign results from the provided training dataset, it can be observed that no individuals from cluster 7 responded to the campaign, whilst most individuals from cluster 0 responded. This fits with the previous analysis where cluster 7 where the most under represented and cluster 0 where the most over represented. This indicate the clustering results are plausible.

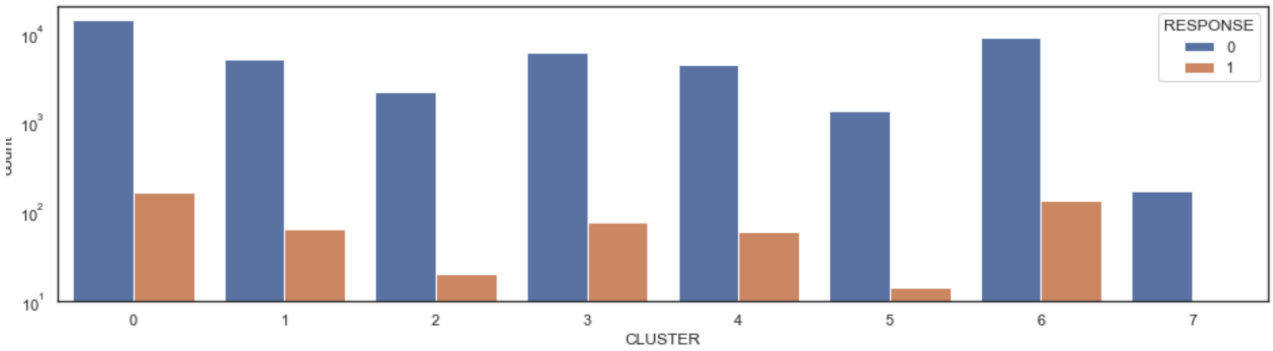


Fig 3.18

Cluster feature was added to the mailout training and testing datasets and then used in the prediction, by using 'LNR' as a key to merge customer and mailout data.

As XGBoost performed the best whilst setting a baseline, this algorithm was selected to further refine and improve the initial baseline set at 64% AUC.

The hyper-parameters that have the greatest effect on optimising the XGBoost evaluation metrics are:

- alpha, min_child_weight, subsample, eta, and num_round, lambda

Description of most important parameters used to solve this problem:

- **alpha:** L1 regularization parameter good for feature selection as it throw away useless data by setting their weights to 0. Increasing this value makes model more conservative [0-1]. As we have over 400 features, this parameter are extremely important
- **lambda**: L2 regularization parameter which shrink least important prediction close to zero, but does not eliminate any features. Increasing this value makes model more conservative
- **booster:** gbtree, gblinear, or dart
- **eta**: learning rate, parameter can be set to control the weighting of new trees added to the model. Smaller numbers avoid overfitting but then we need more trees/estimators
- **gamma**: Minimum loss reduction required to make a further partition on a leaf node of the tree. The larger, the more conservative the algorithm is.

Refinement

AWS Sagemaker was selected to do the final hyper-parameter tuning to improve model accuracy. To set a good base for the initial hyper-parameters, further local manual experiments with XGBoost was done, playing around with some hyper parameters to get a first feel for how the algorithm and hyper-parameters work together. It was observed that when the TRAIN AUC goes over 90%, the model overfits and the VALIDATION score starts to decrease again. Best results are obtained keeping the TRAIN AUC below 90%, this is stops the model from overfitting.

```
[58]    eval-auc:0.75921        train-auc:0.87298
[59]    eval-auc:0.76007        train-auc:0.87426
[60]    eval-auc:0.76020        train-auc:0.87489
[61]    eval-auc:0.76001        train-auc:0.87552
[62]    eval-auc:0.76089        train-auc:0.87638
[63]    eval-auc:0.75928        train-auc:0.87700
[64]    eval-auc:0.75993        train-auc:0.87785
```

In the figure 3.19 on the left we can that train AUC is below 90% and eval_auc is slowly increasing.

```
[500]   eval-auc:0.73742        train-auc:0.98558
[501]   eval-auc:0.73717        train-auc:0.98562
[502]   eval-auc:0.73717        train-auc:0.98562
[503]   eval-auc:0.73717        train-auc:0.98565
[504]   eval-auc:0.73685        train-auc:0.98569
[505]   eval-auc:0.73722        train-auc:0.98566
[506]   eval-auc:0.73709        train-auc:0.98567
[507]   eval-auc:0.73708        train-auc:0.98570
[508]   eval-auc:0.73708        train-auc:0.98570
[509]   eval-auc:0.73708        train-auc:0.98570
[510]   eval-auc:0.73685        train-auc:0.98572
```

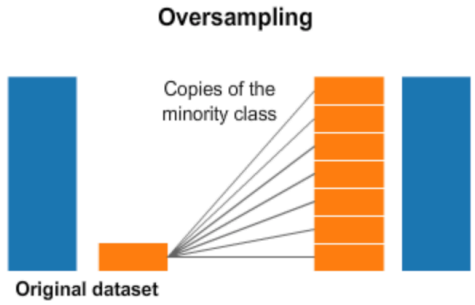
We can now observe when train-arc are over 90%, eval-auc is decreasing slowly

Fig 3.19

To keep TRAIN AUC below 90%, whilst achieving validation AUC around 75%, below values were chosen for hyper-parameters after several observations.

- quite high values for for L1 regulation between 320 and 380 was needed
- learning rate needed to be kept quite low at around 0.1 to avoid overfitting but then more trees/estimators are needed. 5000 estimators were chosen with early stopping rounds of 200. This slowed down the training however seems to be the best way to get AUC above 77%
- setting L2 regulation around 10 is pushing the validation AUC up beyond 76%, but only slightly
- gamma was set at 10 and controls minimum loss reduction required to make a further partition on a leaf node of the tree. This controls tree depth.
- Minimum child weight set at 10 which stops a tree from branching
- max_delta_step are important when dealing with extremely imbalanced datasets/. What max_delta_steps do is to introduce an 'absolute' regularization capping the weight before apply eta correction. This an extra penalization and did seem to make a difference..
- tree depth: if tree depth goes deeper than 4, the model over trains. Best performance are observed with tree depths 2 or 3

As the dataset are extremely imbalanced, further experiments were made using imbalanced-learn package. SMOTE, ADASYN and RandomOverSampler methods was used, of which, surprisingly, the later gave the best results. It even out performed XGBoost's 'scale_pos_weight' parameter.



Over sampling is randomly copying data from the minority class and adding it to the dataset. This makes the minority class to have the same amount of records as the majority class.

Data is not changed, only duplicated.

Fig 3.20

Locally, XGBoost gave improved results of around 76.8% AUC, when tuning parameters as mentioned above.

SageMaker hyperparameter tuning job was defined by setting base parameters as described above

Below hyper-parameters and ranges we selected to tune further. Sagemaker did not seem to perform well when too wide ranges or too many parameters were given. Giving it ranges that are somehow pre-validated and fit together, helped the algorithm to train better and find the most optimal parameters easier.

```
# Set hyperparameter ranges
hyperparameter_ranges = {
                        'eta': ContinuousParameter(0.1, 0.2),
                        'min_child_weight': ContinuousParameter(8,20),
                        'max_depth': IntegerParameter(1, 3),
                        'alpha': ContinuousParameter(320, 380),
                        'lambda': ContinuousParameter(8, 30), # L2
                        'gamma':ContinuousParameter(10, 30),
                        'max_delta_step': ContinuousParameter(8, 20)
                        }
```

Batch transformation was used to predict the results after training, as setting up an end-point for real time prediction is not needed in this use-case.

Other experiments conducted

Below experiments were conducted, however this did improve the best score and thus removed from the final solution:

- Using DART booster with drop-out parameter tuning
- Metric f1-score instead of auc
- Using scale_pos_weight to further control imbalance
- Using PCA components for supervised learning, but this resulted in very low AUC in the low 60%

Challenges to overcome

- One-hot encoding resulted in different columns in the different different datasets as not all values were present in all datasets, one such example was GEBAEUDETYP_5 which was present in full population dataset but not in the customer dataset.
- To one-hot encode or not for ordinal values ? It seemed that behaviour and results are different when fields are one-hot encoded or not which are a bit strange. Separate configuration files for columns were created for supervised vs unsupervised to allow for flexibility.
- It was difficult to figure out how to perform feature analysis with SageMaker models in version 1.3. AWS changed default model output from '.csv' to '.json' and not a lot of documentation existed for this version.
- It was difficult to interpret PCA components when some ordinal features was ranked low to high and others high -> low.
- It seemed better to ohn some ordinal features to better interpret the outcome, although it does not impact performance.
- It's better to not drop any rows in the customer base or mail-out datasets even if they contain over 70% missing data. As kaggle competition is expecting also these almost empty rows, it's better if the training process learn how to handle them, but they can cause some bias. As clustering models are trained using full population dataset with these missing rows dropped, it seems executable.

IV. Results

Model evaluation and validation

The best performing model during sagemaker hyper parameter tuning got a result of 77.24 AUC% with below hyper parameters on the validation dataset.

Best training job hyperparameters

Name	Type	Value
_tuning_objective_metric	FreeText	validation:auc
alpha	Continuous	352.751751519556
booster	FreeText	gbtree
colsample_bytree	Continuous	0.8892962855538444
early_stopping_rounds	FreeText	500
eta	Continuous	0.10112601994782247
gamma	Continuous	27.05383170067603
lambda	Continuous	27.095027308717636
max_delta_step	Continuous	18.40263424550596
max_depth	Integer	2
min_child_weight	Continuous	8.127220163589241
num_round	FreeText	5000
objective	FreeText	binary:logistic
seed	FreeText	88
subsample	FreeText	0.8298039038786528
tree_method	FreeText	exact
verbosity	FreeText	3

Fig 4.1

Using the validation dataset to predict and analyse the results, it is observed that the accuracy of the model are quite low at 71.6%.

Analysing the confusion matrix of the validation dataset, we conclude that out of the 160 individuals who responded, 116 were corrected identified by the model. This is acceptable.
It can also be observed that out of the 12,729 individuals who did not respond, 3616 was incorrectly classified. This are still acceptable, as those 3616 might be possible candidates to target using different strategies in future.

```
Accuracy: 0.7160369307161145

Confusion matrix :
[[9113 3616]
 [  44  116]]

AUC: 0.7738981852462881
```

Feature importance, using total gain

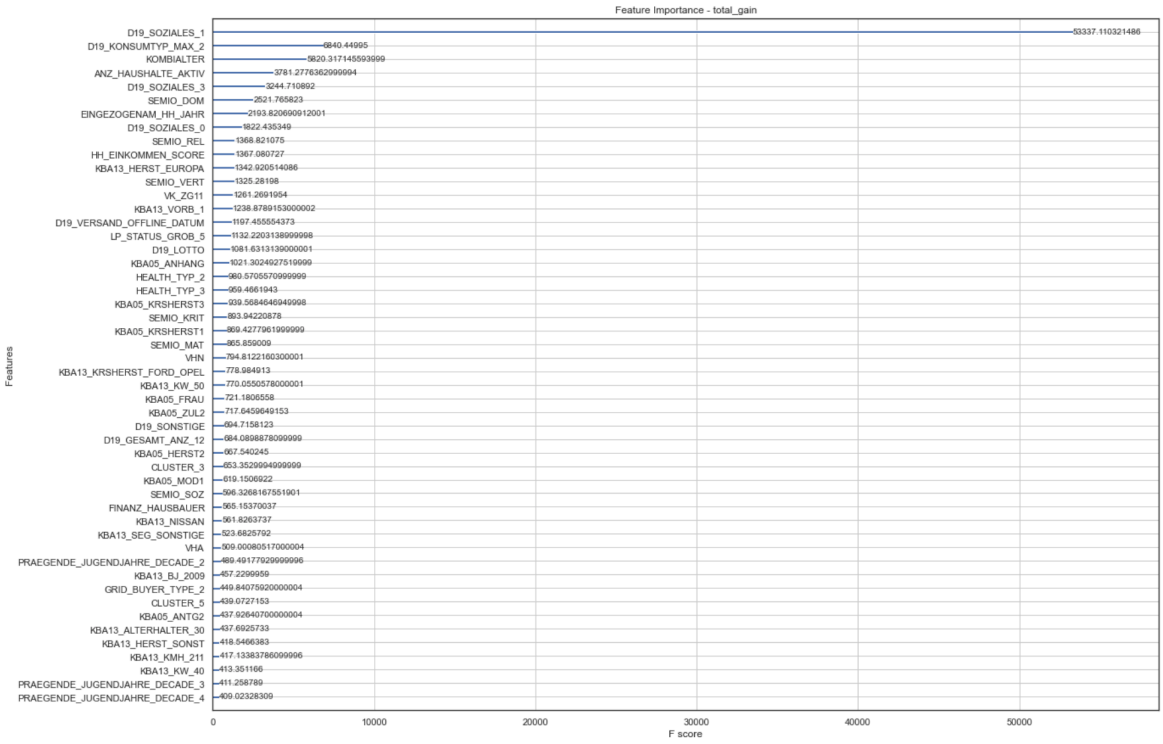


Fig 4.2

Gain is the improvement in accuracy brought by a feature to the branches it is on. The Gain is the most relevant attribute to interpret the relative importance of each feature.

The most important features as seen in Fig 4.2 are D19_SOZIALES_1, D19_KONSUMTYP_MAX_2 and KOMBIALTIER are not described in the data provided.
D19_SOZIALES_1, using intuition, might describe if a person is active on social media.
D19_KONSUMTYP_MAX_2 can be inferred from D19_KONSUMTYP which are consumption type, of which 'Gourmet' seems to be the most dominant after 'Universal' and 'Versatile'
'ANZ_HAUSHALTE_ACTIVE' describe the households in the building
I suspect KOMBIALTIER is related to age due to it's strong correlation with GEBURSJAHR

Visualising tree 0 of 800 gives a deeper understanding how the end-solution could look like. Over 800 such trees were created, with maximum depth of 2. Tree complexity are quite high with 800 trees.

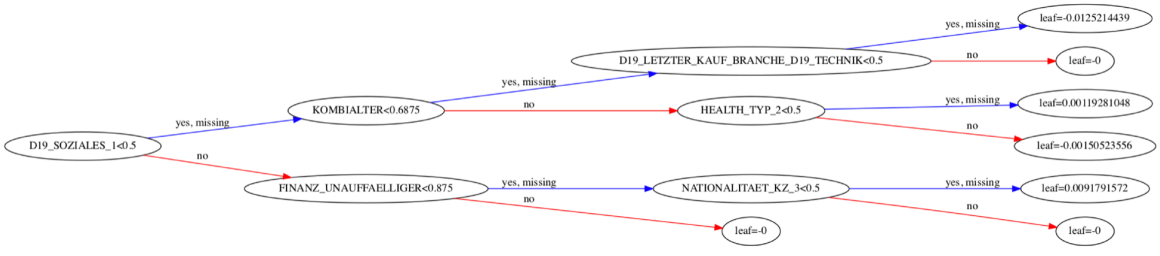


Fig 4.3

Justification

The final result of 79.9% AUC on the testing dataset in kaggle competition was achieved, and is improving the original benchmark score of 64% by 15%. As not a lot of students score over 80%, I think this is a fantastic result giving the complexity and quality of the dataset.

A private score of 76.3% AUC were achieved when only 70% of the testing set was used, which indicate the model has a consistent precision. I would have been 11th position of 438 students if I made the competition deadline.

Submission and Description	Private Score	Public Score
kaggle.csv just now by Juanita Smith	0.76349	0.79852

Final addition of setting colsample_bytree = 0.8, was not improving the final result, but feature importance gave less emphasis to vehicle details which could be coming from those rows where over 60% of values are missing. It is removing the bias and produce more explainable results.

Most important features present in the best performing supervised model, are also similar to features that are present in the most over and under presented clusters and PCA components during unsupervised training, which mostly try to use brands of cars, financial status, income scores to distinguish mostly between low and high income groups. Features describing if the individual is active online seems to come out on top as well.

Cluster 6 seems to be an important feature during supervised learning, which is the 2nd most over represented group in Germany who drive top to middle class cars and live in city centres. This is a bit surprising I would expect cluster 0 and 7 to feature instead.

V. Conclusion

We can now answer the questions defined in the problem statement:

1. Help the mail order company to understand who their customers are

Main customer base are coming from cluster 0, who are presenting high income families living in less dense residential area's possibly in rural area's, or individuals who drive top german car manufacturer brands living in densely populated city centres.

2. How does their client base compare to the rest of Germany ?

Rest of Germany have an even distribution between the 8 different clusters, where the client base of the mail order company are over represented by high income groups, and under represented by young families of low income.

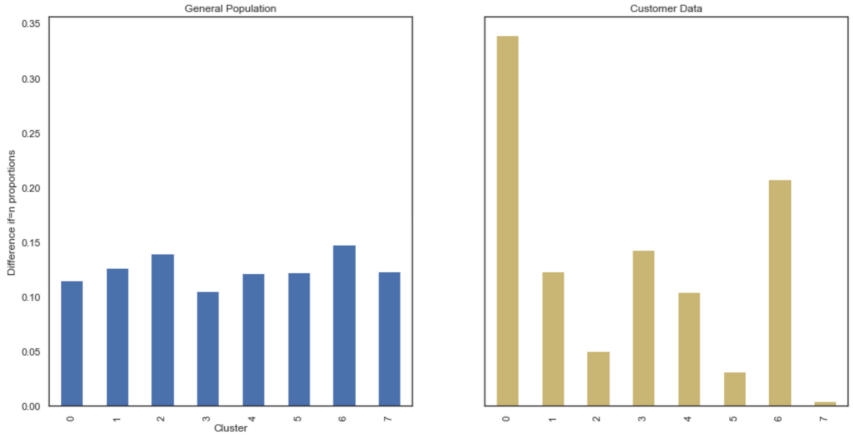


Fig 5.1

3. How can the mail order company acquire new clients more efficiently ?

All persona's in top 5 PCA groups explaining around 25% of the population variance are actually active online customers, even if they are from low income groups and from all age groups. We are living in a digital age, where most individuals do online shopping, just that the mail-order company is not attracting those individuals from lower income groups, especially young families just starting out as described in the most under represented cluster 7 who did not respond at all to the campaign.
The mail order company should consider to offer more competitive products and focussed targeting of individuals in cluster 7 as they do seem to love shopping after all and are searching for great offers.

Fig 5.2

Refer to PCA component 2 in Fig 3.9 as reminder of the full persona description.

Improvements

When saying that data cleaning and data exploration are 80% of data science projects, this project proofed that fact. Without spending the time to analyse and clean the dataset properly, it was difficult to push AUC over 75%. More could be done to remove unused features from the data to reduce model complexity.

Working with AWS sagemaker was quite challenging when one goes beyond Udacity classroom training content, e.g. debugging training, feature analysis, etc.

Use XGBoost feature weights parameter to influence the importance of features like class, and those features playing a role in the most over and under represented clusters. Sagemaker does not seem to support this feature yet, it needs to be investigated.

Investigate how a neural network solution like PyTorch would perform in comparison to XGBoost

Explore deeper the use sampling to overcome data imbalance issues

References

I have completed the first term of this project in nanodegree 'Introduction to Machine Learning using Pytorch', therefore I am familiar with the first part of this project, from the unsupervised learning project. I have copied some of my own previous work to use as a base. The versions of those two datasets used in this project include many more features and has not been pre-cleaned, a lot of additional analysis and coding was needed

Imputing missing values
- https://machinelearningmastery.com/iterative-imputation-for-missing-values-in-machine-learning/
- https://scikit-learn.org/stable/modules/generated/sklearn.impute.IterativeImputer.html
- https://stackoverflow.com/questions/61945250/im-getting-negative-values-as-output-of-iterativeimputer-from-sklearn

Imbalanced datasets:
- https://towardsdatascience.com/how-to-deal-with-imbalanced-data-34ab7db9b100
- https://towardsdatascience.com/classification-framework-for-imbalanced-data-9a7961354033
- https://imbalanced-learn.org/stable/references/over_sampling.html#smote-algorithms

- https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/
- https://medium.com/analytics-vidhya/how-to-handle-imbalanced-dataset-b3dc05b85bf9
- https://imbalanced-learn.org/stable/auto_examples/over-sampling/plot_shrinkage_effect.html

XGBoost:
- https://towardsdatascience.com/xgboost-is-not-black-magic-56ca013144b4
- https://xgboost.readthedocs.io/en/latest/python/python_intro.html#training
- https://machinelearningmastery.com/tune-learning-rate-for-gradient-boosting-with-xgboost-in-python/
- https://github.com/dmlc/xgboost/blob/master/demo/guide-python/sklearn_examples.py
- https://www.kaggle.com/cast42/xgboost-in-python-with-rmspe-v2
- https://github.com/dmlc/xgboost/blob/master/demo/guide-python/feature_weights.py
- https://discuss.xgboost.ai/t/feature-weights-does-not-work-as-expected/1998/3

Feature Selection:
- https://towardsdatascience.com/how-to-use-variance-thresholding-for-robust-feature-selection-a4503f2b5c3f

PCA
- https://online.stat.psu.edu/stat505/lesson/11/11.4
- https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues

AWS and Sagemaker
- https://aws-ml-blog.s3.amazonaws.com/artifacts/prevent-customer-churn/part_2_preventing_customer_churn_XGBoost.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/debugger-training-xgboost-report.html
- https://machinelearningmastery.com/tune-learning-rate-for-gradient-boosting-with-xgboost-in-python/

Metrics:
- https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
- https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781838555078/6/ch06lvl1sec34/confusion-matrix

Visualisation
- https://stackoverflow.com/questions/49564844/3d-pca-in-matplotlib-how-to-add-legend
- https://stackoverflow.com/questions/28227340/kmeans-scatter-plot-plot-different-colors-per-cluster

Environment setup
- https://stackoverflow.com/questions/35802939/install-only-available-packages-using-conda-install-yes-file-requirements-t
- https://subscription.packtpub.com/book/data/9781800208919/2/ch02lvl1sec06/setting-up-amazon-sagemaker-on-your-local-machine
- https://aws.amazon.com/premiumsupport/knowledge-center/sagemaker-lifecycle-script-timeout/