# Capstone Project: Customer Segmentation Report for Arvato Financial Services

Machine Learning Engineer Nanodegree

*Shenzhen Lu*

*March 11st, 2020*

# I. Definition

## Project Overview

Customer segmentation allows marketers to better tailor their marketing efforts to various audience subsets. Those efforts can relate to both communications and product development. In B2C marketing, companies often segment customers according to demographics.

Traditional customer analyses are often quite inefficient and less transparent, the idea of leveraging machine learning techniques to automate the customer segmentation process has been gaining popularity in the era of “Big Data”: It doesn’t take a lot efforts to find out quite amount of papers and articles about this domain in search engine.

In this project, we will analyze demographics data for customers of a mail-order sales company in Germany, comparing it against demographics information for the general population. We'll use unsupervised learning techniques to perform customer segmentation, identifying the parts of the population that best describe the core customer base of the company. Then, we'll apply supervised learning algorithms on a third dataset with demographics information for targets of a marketing campaign for the company and predict which individuals are most likely to convert into becoming customers for the company. In order to achieve these goals, we will primarily utilize the Python libraries Pandas and Scikit-Learn in this project.

The data that we will use has been provided by Bertelsmann Arvato Analytics, and represents a real-life data science task. There are four data files associated with this project:

- *Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany*
- *Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company*
- *Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign*

- *Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign*

# Problem Statement

Our project consists of three main parts: customer segmentation report, supervised learning model and Kaggle competition.

## Customer Segmentation Report

The main bulk of your analysis will come in this part of the project. Here, we use unsupervised learning techniques to describe the relationship between the demographics of the company's existing customers and the general population of Germany.

1. Concatenate AZDIAS and CUSTOMERS data
2. Data cleaning and preprocessing
3. Dimensionality reduction with PCA on numerical data
4. Cluster transformed data with K-Means
5. Analyze clusters to describe which parts of the general population are more likely to be part of the customer base, and which parts of the general population are less so
6. Analyze the categorical features to achieve the same goal as above

## Supervised Learning Model

Now that we've found which parts of the population are more likely to be customers of the mail-order company, it's time to build a prediction model.

1. Clean and preprocess the MAILOUT training data
2. Modeling with Logistic Regression and Support Vector Machine

    - Standardize numerical features and create dummy variables out of categorical features

- Feature selection by recursive feature elimination with Logistic Regression model, the best number of features is determined automatically with K-fold cross-validation
- Construct Logistic Regression and Support Vector Machine classifiers and optimize hyperparameters with grid-search cross-validation, refit estimators with the best-found parameters on the whole selected dataset

3. Modeling with Random Forest and Gradient Boosting Trees

    - Features selection of the preprocessed data (step 1) by recursive feature elimination with Random Forest classifier, the best number of features again is determined automatically with K-fold cross-validation
    - Construct Random Forest and Gradient Boosting classifiers, optimize hyperparameters with random-search cross-validation, refit estimators with the best-found parameters on the above dataset

### Kaggle Competition

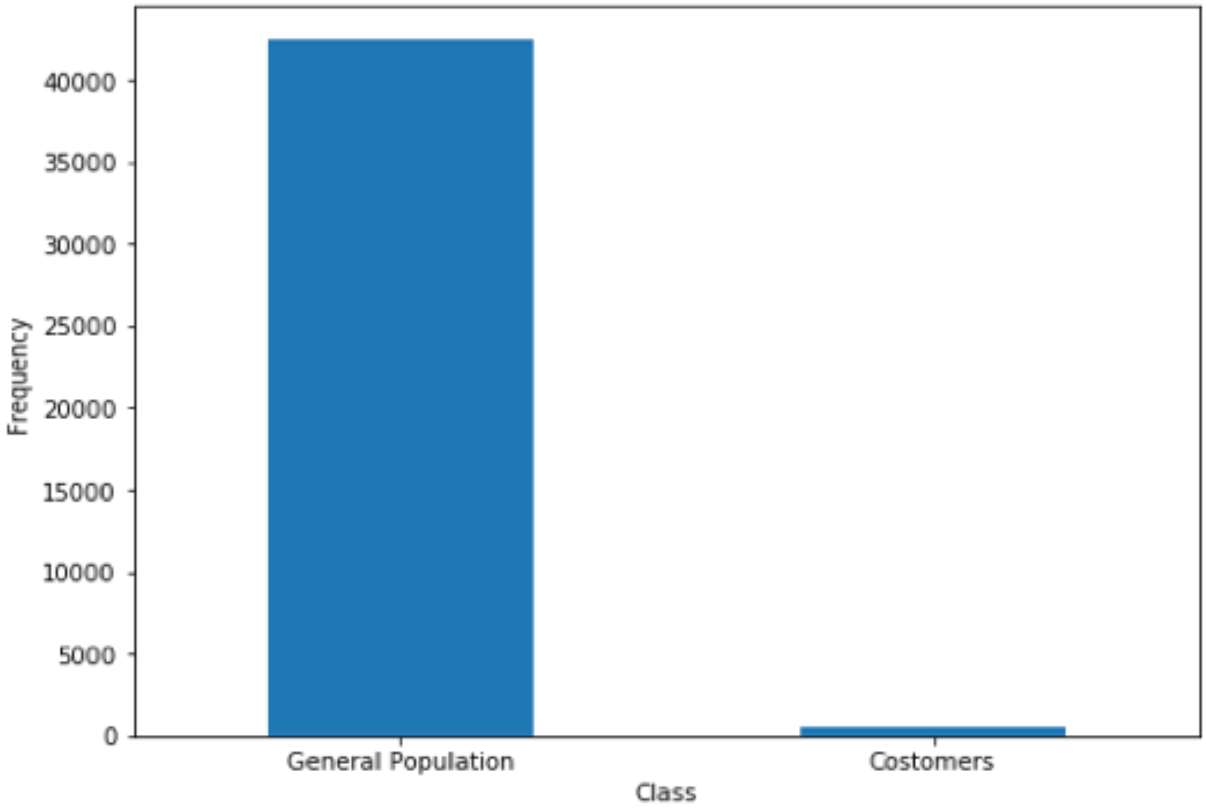
Now that we've created a model to predict which individuals are most likely to respond to a mailout campaign, it's time to test that model in competition through Kaggle.

The steps involved in this part are the following:

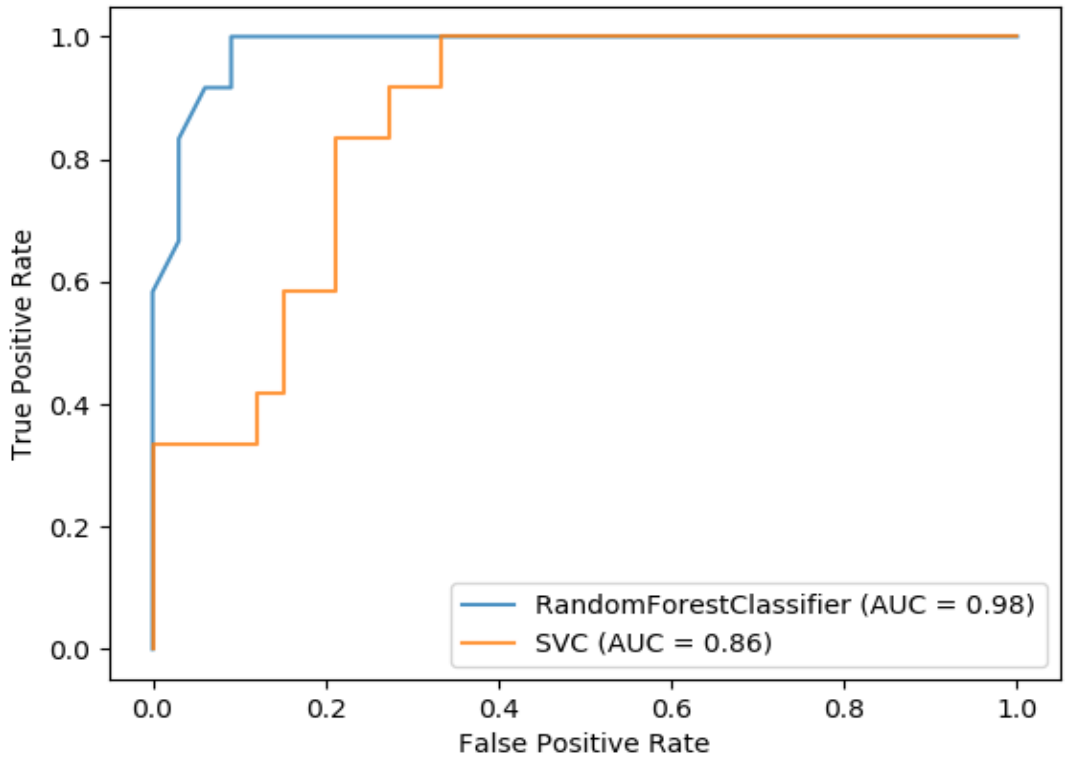
1. Clean and preprocess MAILOUT test data in refer MAILOUT train data
2. Predict probability estimates on test data
3. Upload predictions to Kaggle competition in specified format

## Metrics

There is a large output class imbalance in MAILOUT train data, where most individuals did not respond to the mailout.




Thus, predicting individual classes and using accuracy does not seem to be an appropriate performance evaluation method. Instead, the competition will be using *AUC (Area under the ROC curve)* to evaluate performance: the exact values of the "RESPONSE" column do not matter as much, only that the higher values try to capture as many of the actual customers as possible, early in the *ROC* curve sweep. An example is showed in the figure below [1].




# II. Customer Segmentation Report

## Data Exploration and Cleaning

The data files associated with this part:

- *Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns).*
- *Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns).*

Each row of the demographics files represents a single person, but also includes information outside of individuals, including information about their household, building, and neighborhood.

For more information about the columns depicted in the files, two Excel spreadsheets are given by the dataset provider: one of them (DIAS Information Levels - Attributes

2017.xlsx) is a top-level list of attributes and descriptions, organized by informational category. The other (DIAS Attributes - Values 2017.xlsx) is a detailed mapping of data values for each feature in alphabetical order.

After loading the 2 data files, we first combined them together in one Dataframe, then split it to features and target.

After exploration of the features and additional information files, the following **Problems** have been detected:

- A typo exists by feature *KBA13_CCM_1401_2500*
- Missing value indicators of several features are either not consistent with or can simply not been found in the information files
- A few features in the information files don't exist in the data, many features in the data don't have corresponding descriptions in the information files, some features show up in both positions without consistent names

Thus, we have provided the following **Solutions:**

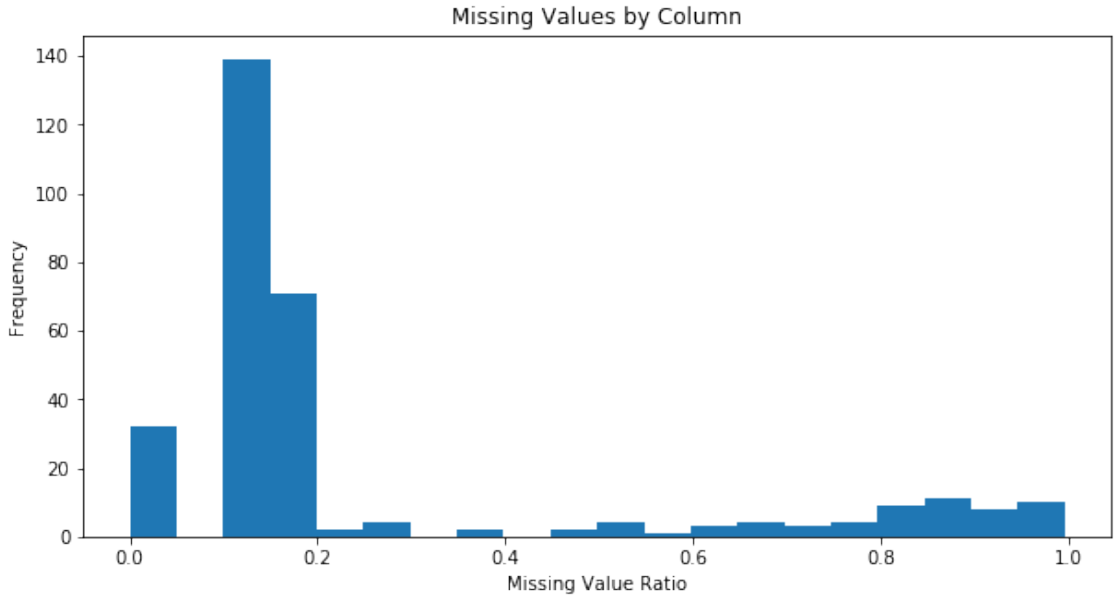
- Correct the typo by renaming feature *KBA13_CCM_1401_2500* to *KBA13_CCM_1400_2500*
- Replace bad missing value indicators with proper value
- Creating a new spreadsheet file (DIAS_features_info.xlsx) for reference: column *Feature* contains names of feature mentioned in the information files except for those not showing up in data, some are modified to consistent with the data; *Description* describes the meaning of features; *NaN_Indicators* represents missing or unknow values; Data_*Type* describes the statistical data type and *Notes* additional comments.
- Replace missing value indicators with np.nan's
- Subset data with features included in the information file

# EDA and Data Preprocessing

## Dealing with Missing Values

In this section, we will choose proper thresholds to remove columns and rows with too many missing values: if the threshold is too loose, we might have many redundant features; if the threshold is too strict, we might have few features to analyze.

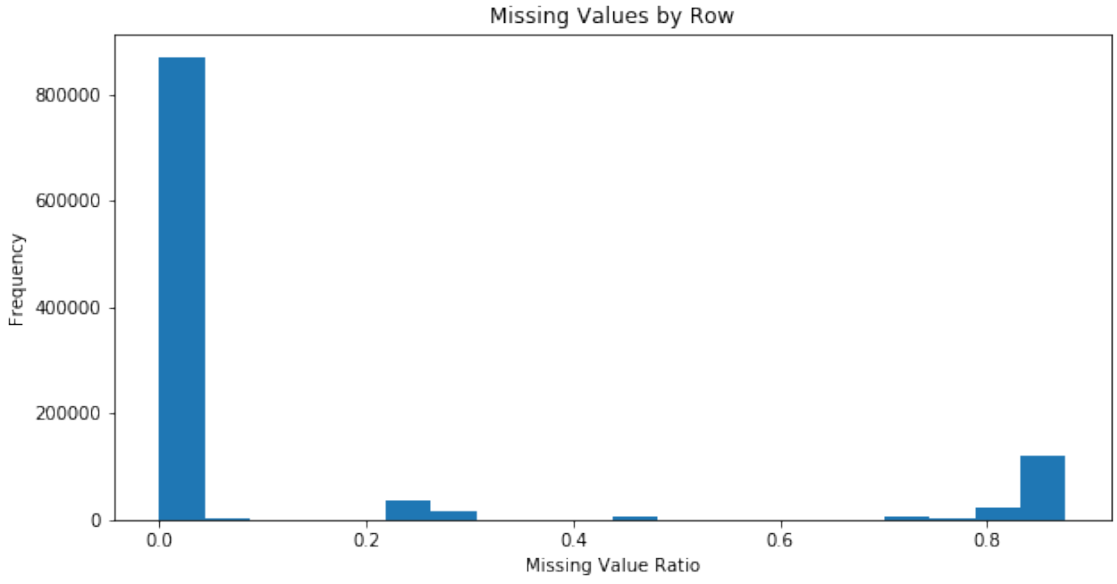
1. Missing Values by Column




As a majority of features has less than 20% missing values, it's reasonable to set it as the threshold, any features with more than 20% missing values will be dropped.

After filtering the columns, we will examine the rows.

2. Missing Values by Row



Most of rows have less than 10% missing values; nevertheless, as more tolerance for rows is allowed, we will choose 40% as the threshold.

## Dealing with Statistical Data Types

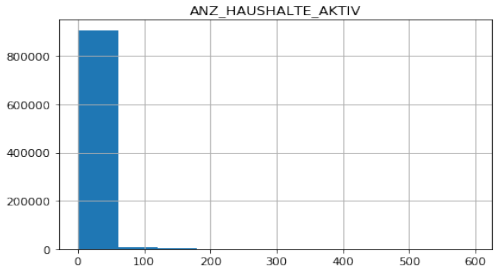
The features consist of 4 statistical data types: Ordinal, Discrete, Mixed and Categorical [2].

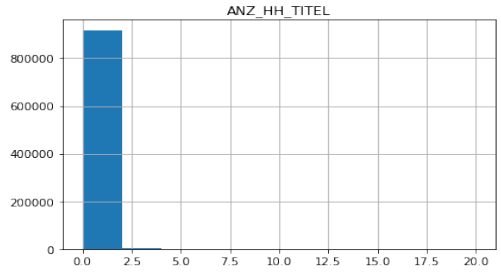
1. Ordinal

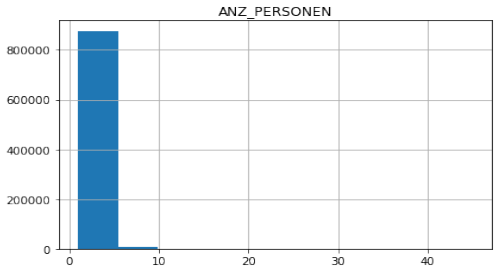
   - Ordinal data has nature, ordered categories and distance between the categories is not known. In this project, ordinal features are represented as integers and treated as numerical data
   - Missing values are imputed with the mean

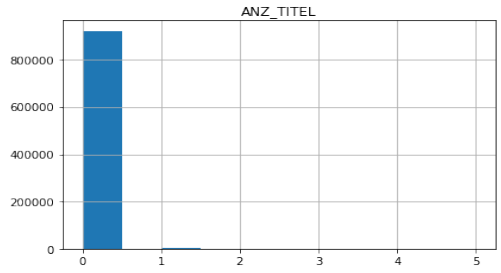
2. Discrete

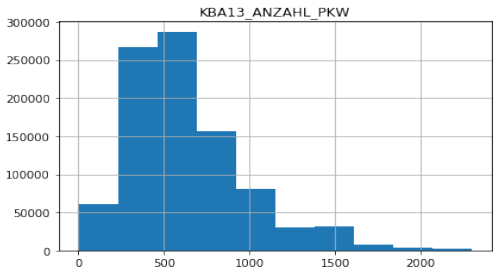
   - Discrete data is numerical data with discrete values, e.g. number of cars in the PLZ8

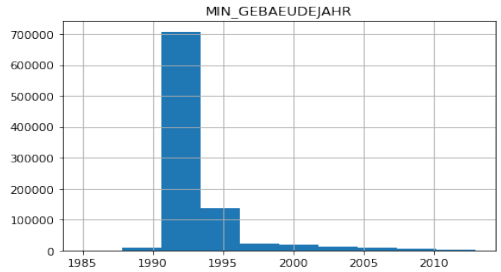


- *ANZ_HH_TITEL, ANZ_PERSONEN, ANZ_TITEL* are dropped due to low variance
- Detect outliers and replace them with np.nan
- Log-transform *ANZ_HAUSHALTE_AKTIV* as the distribution is badly left-skewed
- Missing values are imputed with the mean

3. Categorical and Mixed

   - Unlike ordinal data, categorical data has categories that cannot be ordered or measured
   - Mixed type is the mixture of categorical and ordinal type. In this project, mixed data is treated as categorical data
   - *CAMEO_INTL_2015, LP_LEBENSPHASE_FEIN, LP_LEBENSPHASE_GROB, LP_STATUS_FEIN were dropped* as the information they could provide has been contained in other features

4. Data Transformation

   For further processing, the numerical features need to be standardized: removing the mean and scaling to unit variance. [3]

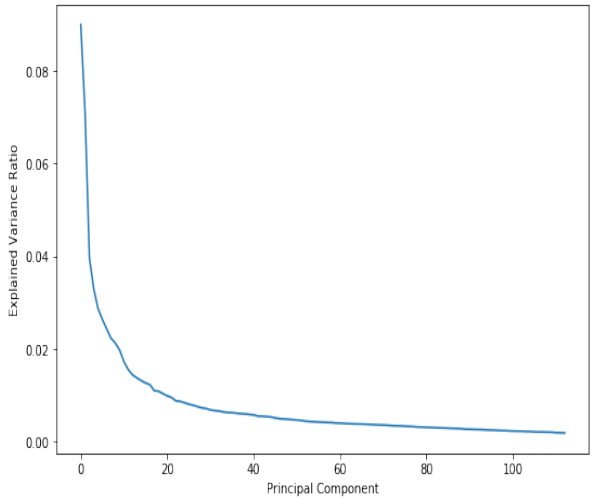
# Algorithms and Implementation

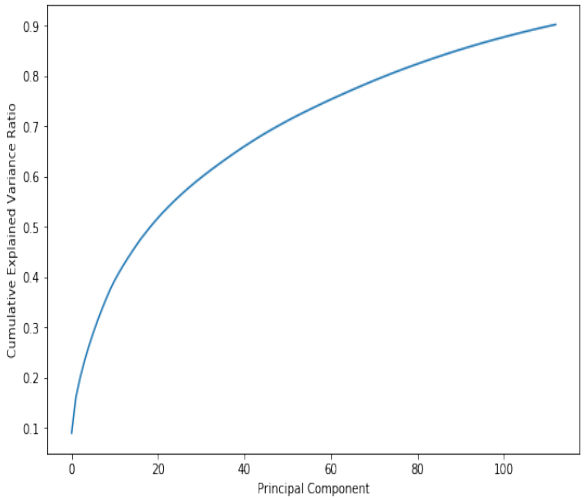
## Dimensionality Reduction with PCA

A large number of features could cause the following problems:

- Too many redundant features
- Long computation time
- The curse of dimensionality

Therefore, Principal Component Analysis [4] was performed to reduce the feature space of the data. Eventually, in order to keep 90% of the variance, 113 Principal Components were extracted from 222 features.

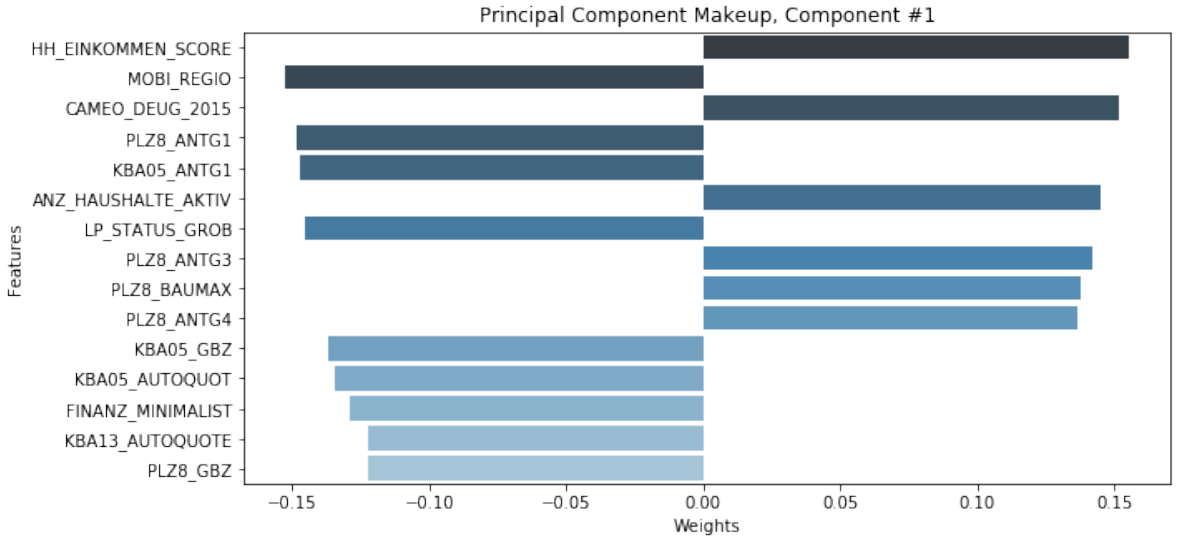



## Principal Components Interpretation

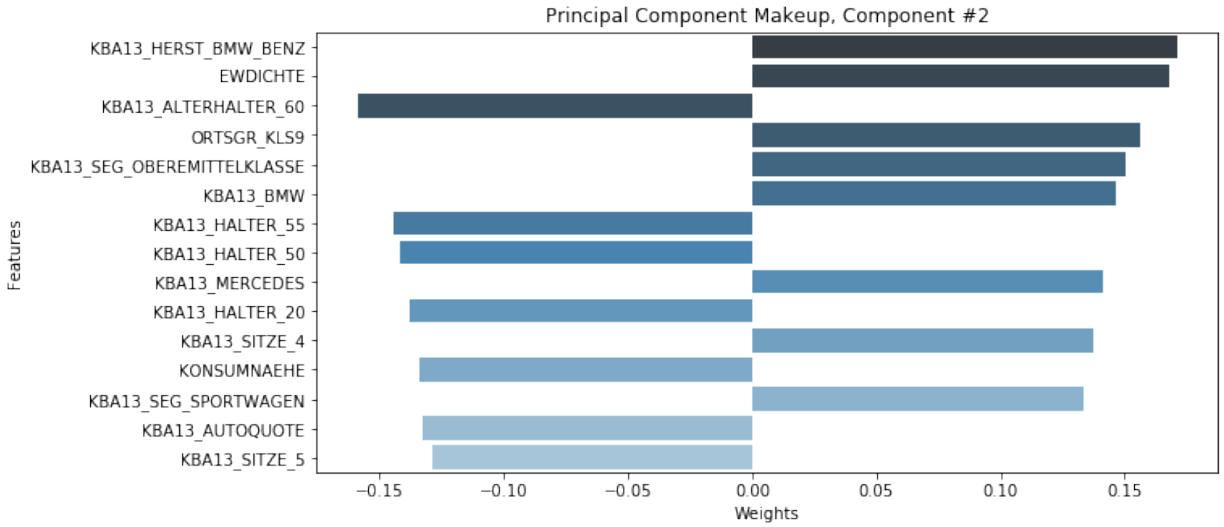
We will examine the highest-weighted makeup features of the Principal Components (PCs) and then come up with a high-level interpretation.

### *1. First Principal Component*



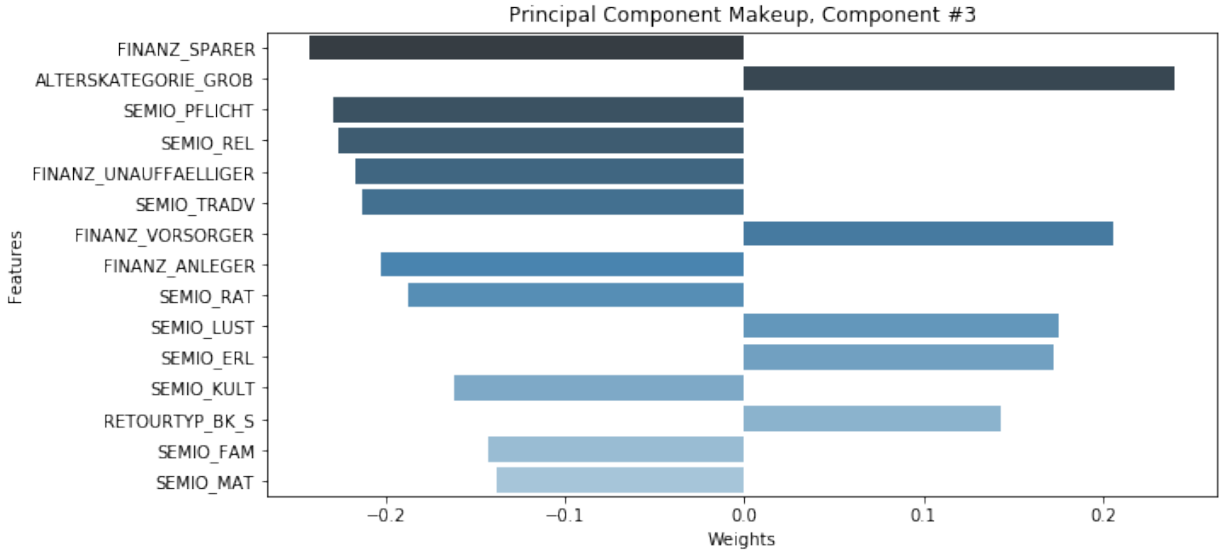

- **Feature Makeup:** low Income, low social class, high mobility (moving often), low financial interest, low share of cars per household, low share of 1-2 family houses and high share of 6+ family houses in the region
- **Principal Component Interpretation:** Lower Class Index

### *2. Second Principal Component*



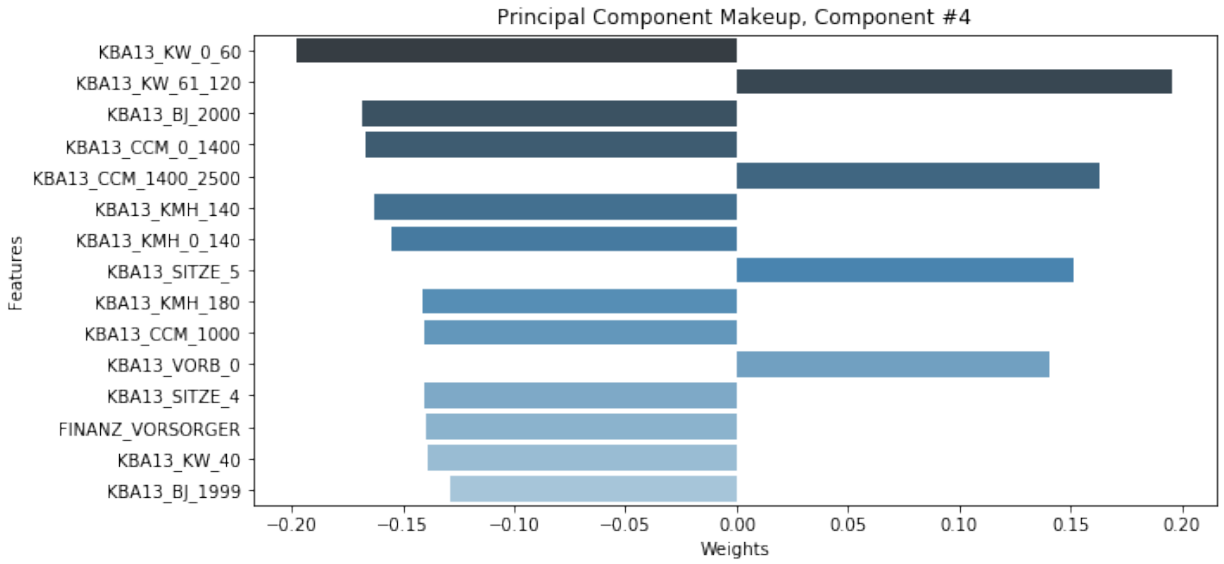

- **Feature Makeup:** large community size, high inhabitant density, high share of upper-class cars as well as cars with less than 5 seats in the PLZ8, few car owners older than 46 and younger than 21
- **Principal Component Interpretation:** Urban Area/Metropolis Index

### *3. Third Principal Component*




- **Feature Makeup:** high savings, elder, wealth management, dutiful / religious / rational / familiar minded
- **Principal Component Interpretation:** Financial Awareness Index

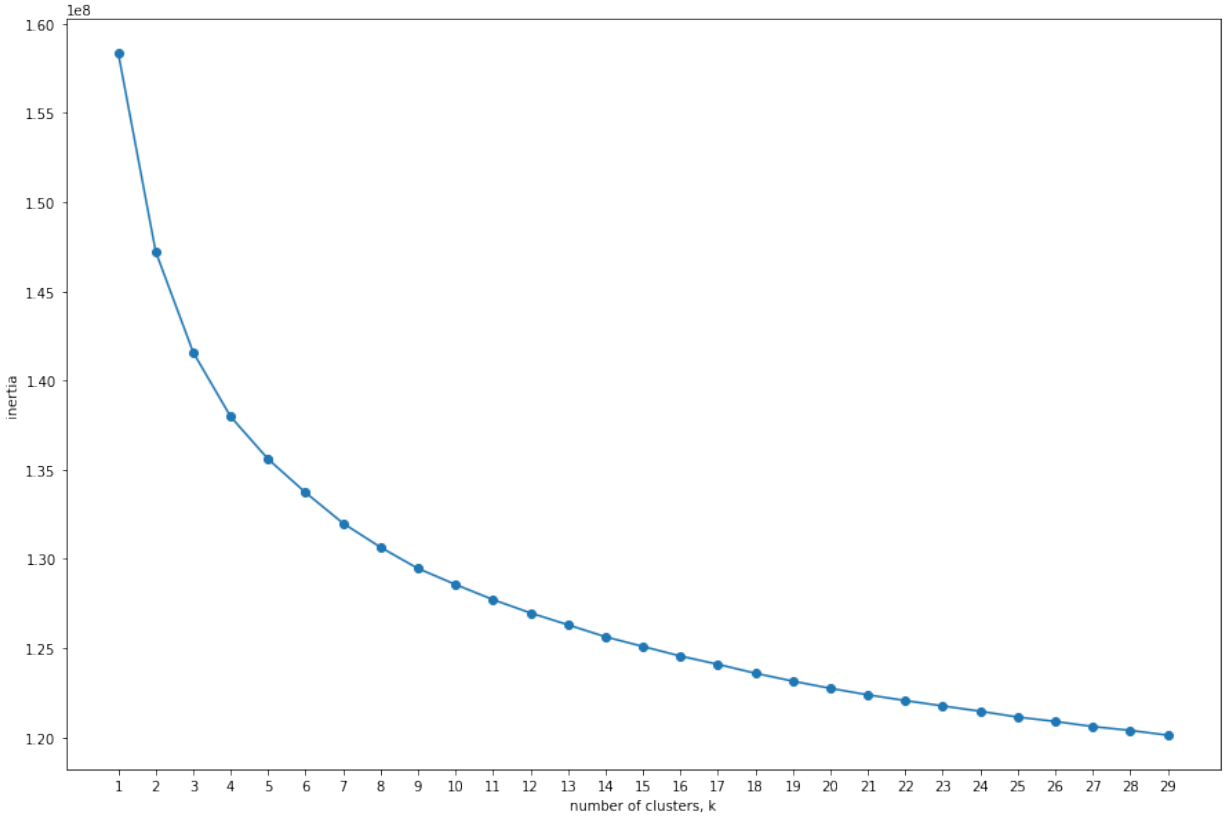
### *4. Fourth Principal Component*




- **Feature Makeup:** Region with low share of cars up to 60 KW engine power, less than 1400ccm, with max speed 140 km/h, less than 5 seats, built before 2003; high share of cars with engine power between 61 and 120 KW, with 1400ccm to 2499ccm, 5 seats; younger car owners.
- **Principal Component Interpretation:** Performant-Car Region Index

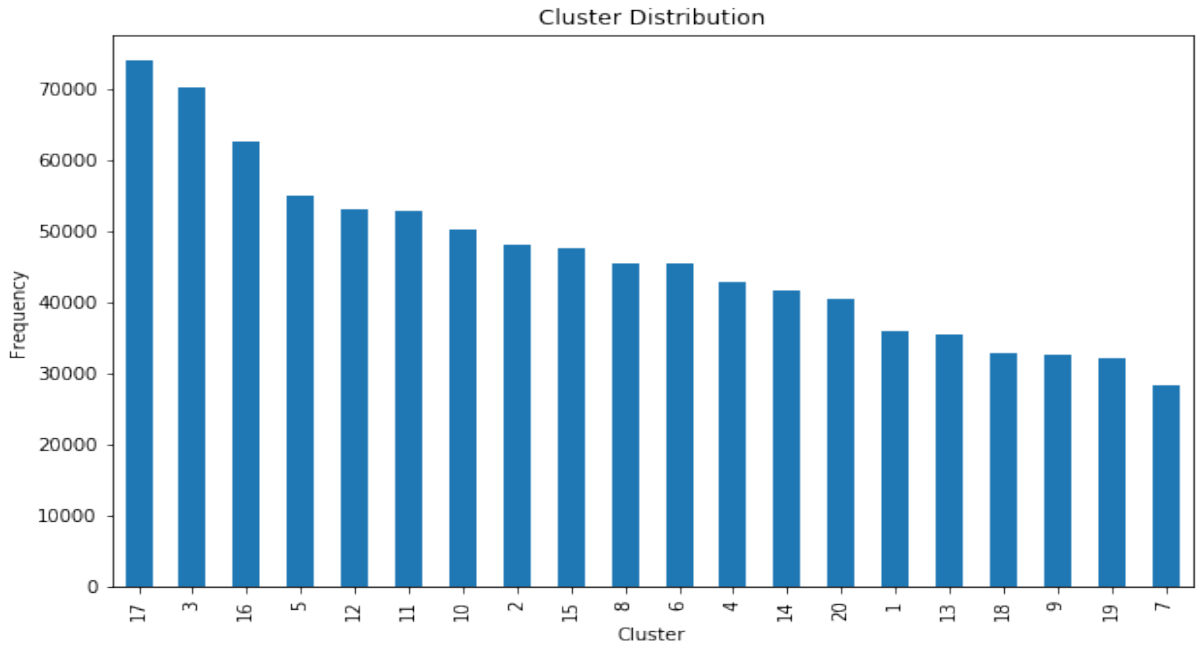
## K-Means Clustering

In this section, we applied K-Means algorithm to cluster the population with similar characteristics [5]. In order to find the optimal number of clusters, the ‘elbow’ method was used.



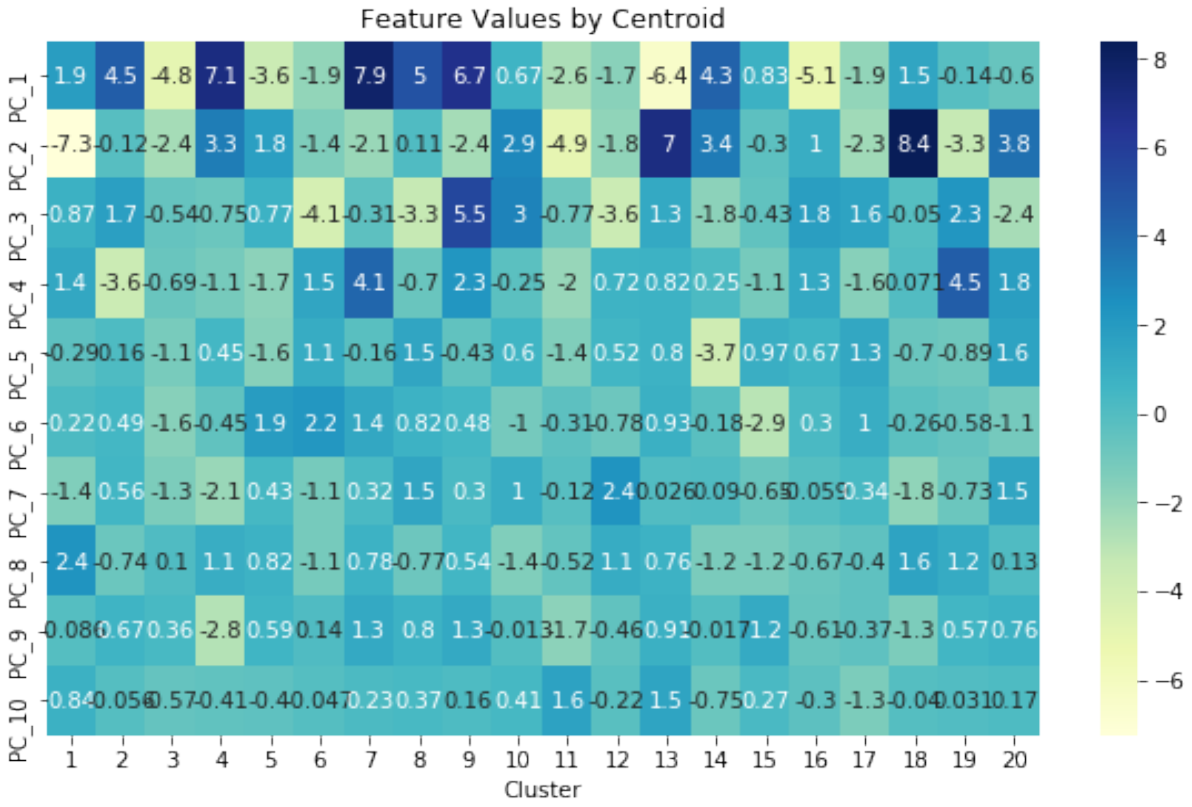

The inertia decreases smoothly as the number of clusters k increases, this implies that the boundary between the clusters may not be quite significant.

Nevertheless, we chose 20 as the optimal number of clusters since the inertia seems to be low enough and decreases very slow after.



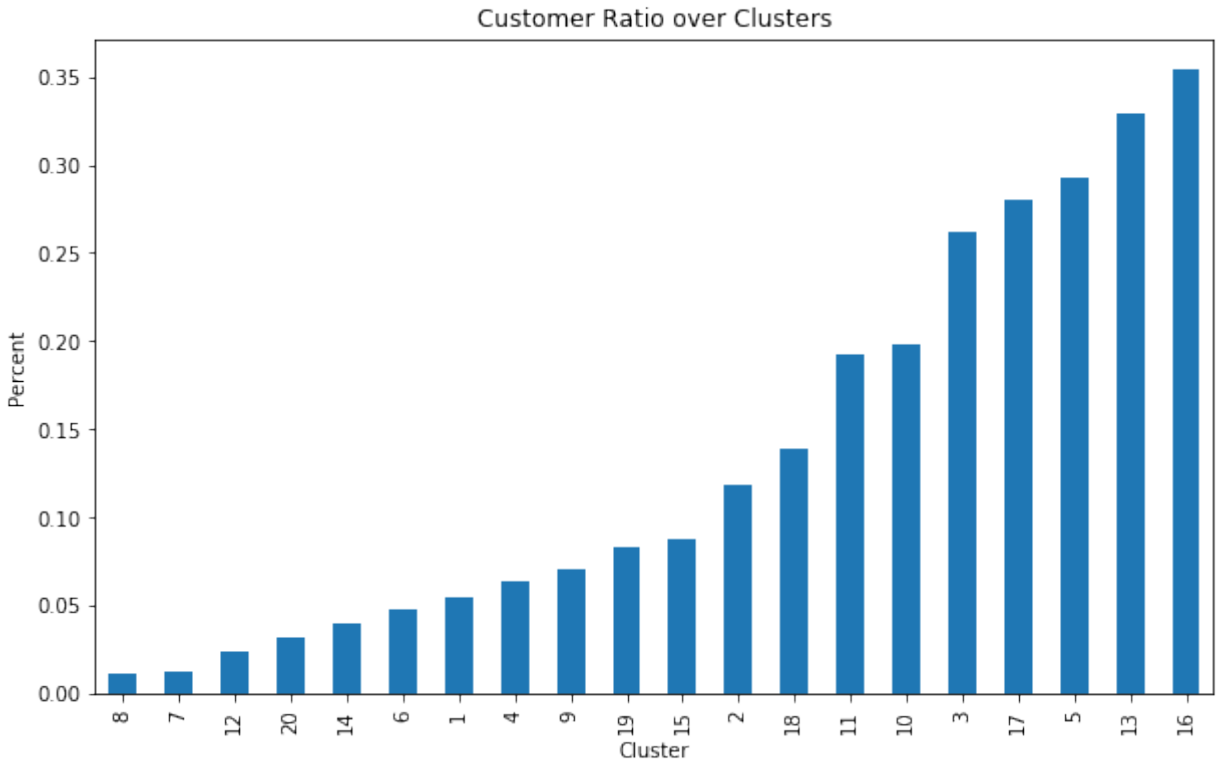
As a matter of fact, the data distributes quite uniformly across the clusters, this a sign of good choice of k.

The clusters are represented by their centroids: the location of the centers of each cluster in feature space. The centroids give us insight into what characteristics (highest-weighted PCs) define each cluster and could be visualized with heatmap plot.



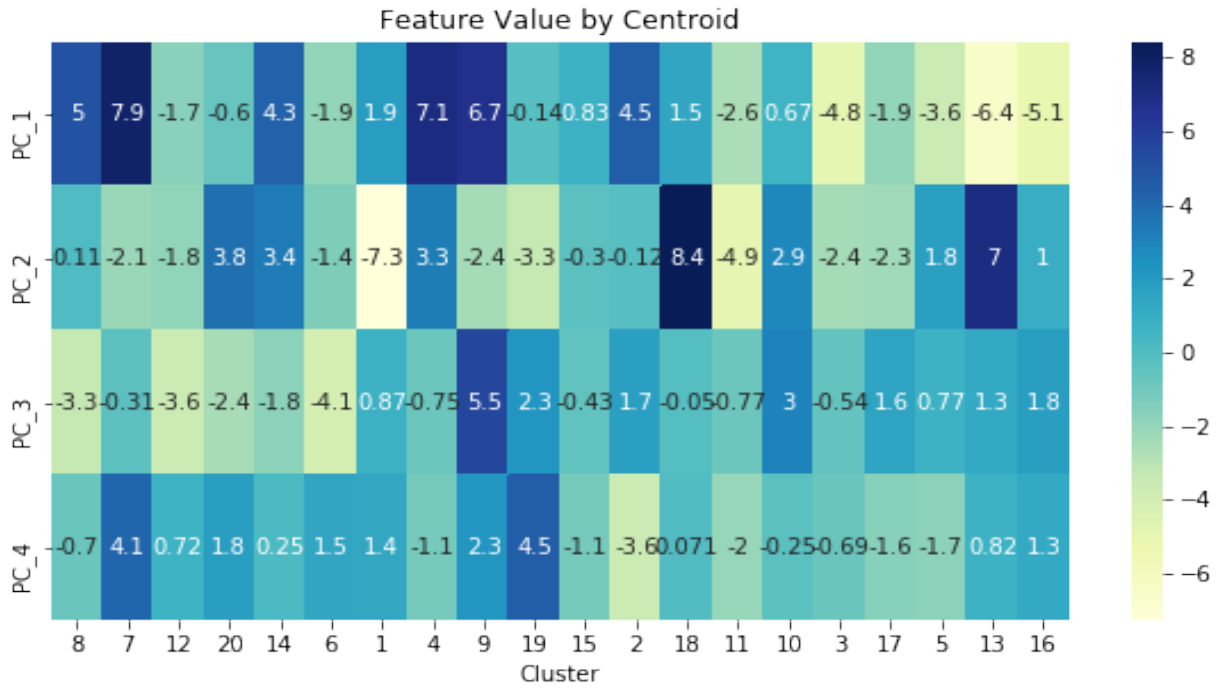

## Cluster Analysis and Customer Segmentation

In this section, we will draw the costumers' profile by comparing characteristics and customer ratio of the clusters: if a cluster has higher customer ratio, then the characteristics of the cluster can better describe the customers.




Clusters 16, 13 and 5 are with the highest customer proportion, while 8, 7 and 12 are the lowest.

As we have noticed above, the clusters' characteristics are dominated by the first 4 Principal Components. Let's check out the heatmap with only those 4 features, the cluster numbers are sorted in ascending order based on the customer ratio.



**Cluster Analysis:**

- PC_1 (lower class index) is significantly negative correlated with customer ratio cross the clusters
- PC_2 (Urban Area/Metropolis Index) doesn't show consistent correlation with customer ratio
- PC_3 (Financial Awareness Index) has positive correlation with customer ratio
- PC_4 (Performance-Cars Region Index) has negative correlation with customer ratio

Finally, we can come up with an overall conclusion of the customer profile:

> ***Our customers have good incomes and high social status; they might live in urban area and metropolis as well as rural area and small towns; they are elder people who save money and have* active *financial investment; they are young people live in regions with high share of small cars.***

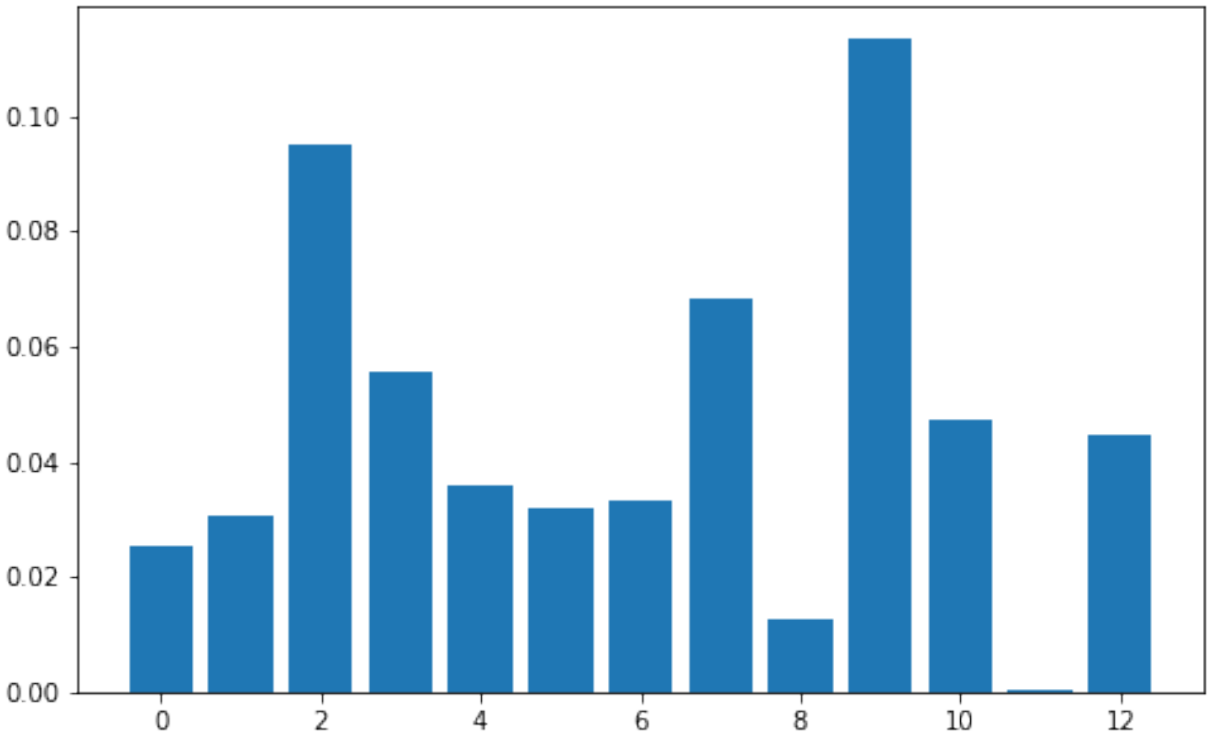
# Customer Segmentation with Categorical Features

Having analyzed the numerical data, let's also have a look at the categorical features, see if they could provide useful information to segment customers from general population.

To analyze all categorical features is tedious and unnecessary, thus, we will only select the most informative ones.

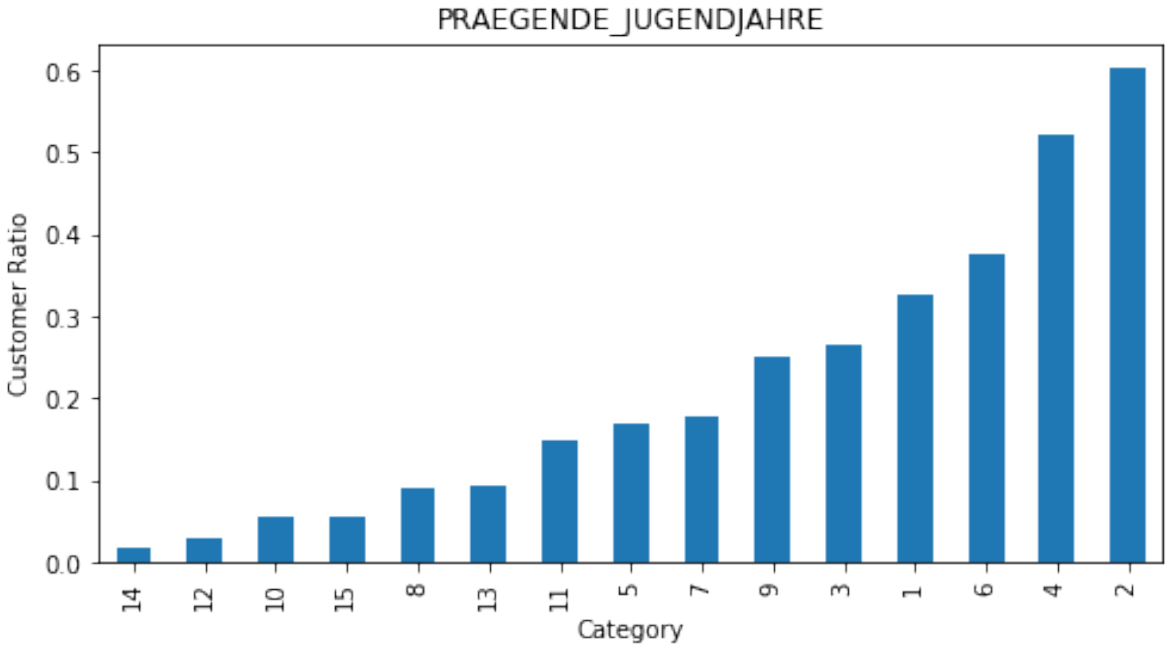
## Feature Selection with Mutual Information

Mutual Information (MI) measures the dependency between two categorical variables., higher value means higher dependency [6]. After calculating the MI scores between categorical features and the target, only the features with highest scores were kept.




2, 3, 7, 9 are the most relevant, these features are *PRAEGENDE_JUGENDJAHRE*, *ANREDE_KZ*, *HEALTH_TYP* and *OST_WEST_KZ*.

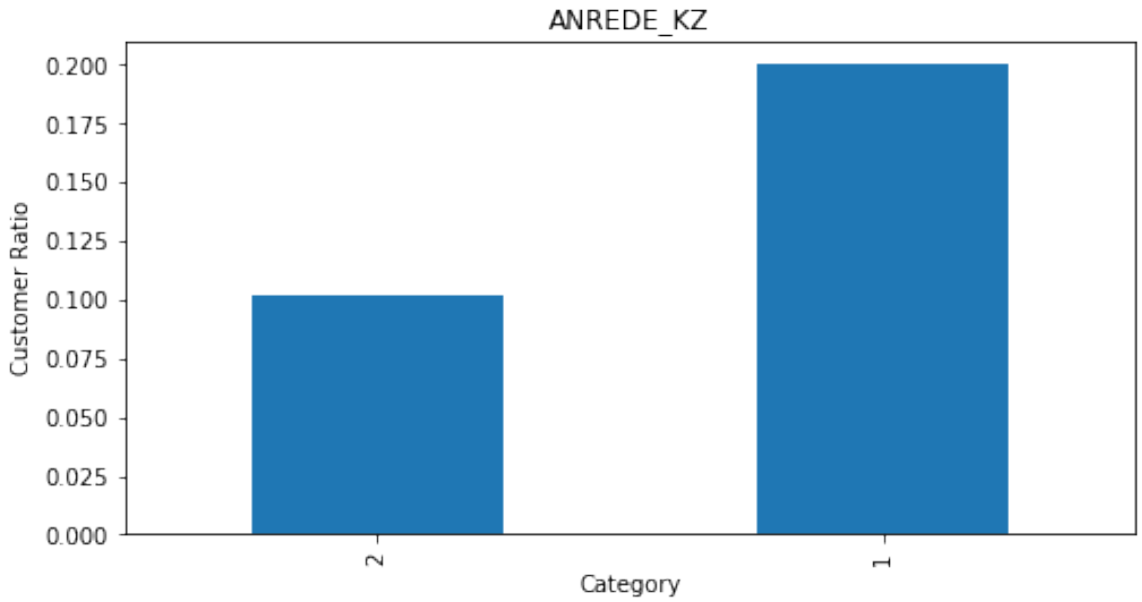
***PRAEGENDE_JUGENDJAHRE***




**Analysis:** In category 2 and 4 the customer ratio is exceptionally high with over 50%, notice that the overall customer proportion is only around 15%. This implies people

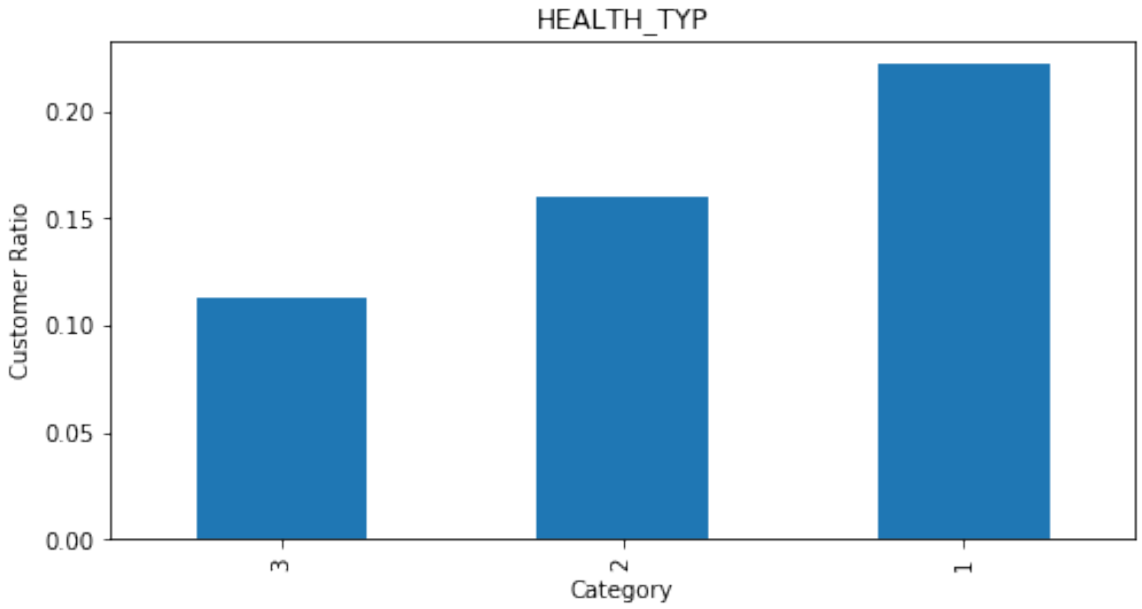
born in 40ies-reconstruction years and 50ies-milk bar/Individualization ages are highly likely to be our customers.

### 1. *ANREDE_KZ*



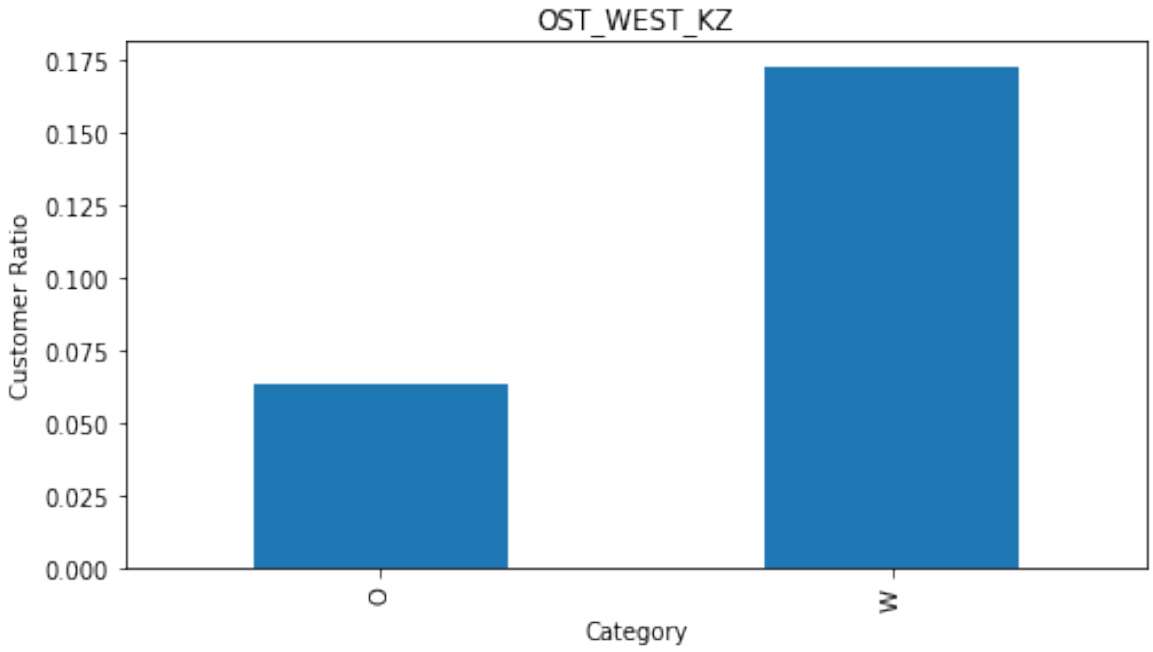

**Analysis:** Category 2 represents female and category 1 male. Obviously, males are more likely to be our customers.

### 2. *HEALTH_TYP*



**Analysis:** Critical-reserved health type (Category 1) is most likely to be our customer, followed by sanitary-affine (Category 2) and aunty-hedonists (Category 3).

### *3. OST_WEST_KZ*




**Analysis:** People live in former West Germany (Category W) are much likely to become our customers than in former East Germany (Category O).

# III. Supervised Learning Model and Kaggle Competition

## Data Cleaning and Preprocessing

There are two data files associated with this part:

- *Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns)*
- *Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns)*

Each of the rows in the MAILOUT data files represents an individual that was targeted for a mailout campaign. We will build and verify our model with the training data, the "TRAIN" partition of MAILOUT data files that includes a column *RESPONSE*, that states whether or not a person became a customer of the company following the campaign. As next, we will create probability predictions on the test data, the "TEST" partition of MAILOUT data files, where the *RESPONSE* column has been withheld. The predictions will be assessed with *AUC* metric in the Kaggle competition.

Unlike in the customer segmentation part, we will include all the features here. Because our main purpose is to achieve good performance not interpretability.

The cleaning process of the train data was identical to the customer segmentation part for those features existed in the feature Information file (DIAS_features_info.xlsx). Otherwise, we assume the missing value indicator be -1 and object-typed feature in Panda Series categorical.

Missing values of numerical features were imputed with the Means and categorical features with the Modes.

# Algorithms and Techniques

1. Choices of machine learning models: **Logistic Regression, Support Vector Machine, Random Forest, Histogram-based Gradient Boosting Tree** [7] [8] [9] [10]
2. Feature Engineering

    - Standardize numerical feature and dummy-encode categorical feature for LR and SVM
    - Encode categorical feature as ordinal integers for RF and HGBT
3. Feature Selection

    - Recursive feature elimination and cross-validated selection of the best number of features [11]
    - Logistic Regression as base estimator for LR and SVM
    - Random Forest as base estimator for RF and HGBT

- Assign weights inversely proportional to class frequencies in the input data to combat class imbalance problem

4. Use Grid Search [12] and Random Search [13] to find the optimal hyperparameters with the best cross validation score (*AUC score*)

| Models | Best CV Score |
|---|---|
| Logistic Regression | 0.709975 |
| SVM | 0.744998 |
| Random Forest | 0.764368 |
| HGBT | 0.772592 |

5. Refit the models with optimal hyperparameters on the whole train dataset

# Kaggle Competition

## Preparation

1. Clean and preprocess the test data in refer to the training data
2. Make predictions (probabilities) on the test data with the trained models
3. Create CSV files with the predictions in required format and upload to Kaggle

## Results

| Models | Kaggle Leaderboard Score |
|---|---|
| Logistic Regression | 0.68445 |
| SVM | 0.74868 |
| Random Forest | 0.79125 |
| HGBT | 0.80053 |

The models' performance on Kaggle is quite consistent with the cross-validation score in terms of ranking. Both HGBT and Random Forest have achieved the goal we set in the proposal report that the test score should lie in range 0.79 - 0.80. Eventually, our best score 0.80053 ranked 42 in the leaderboard, notice that there are 141 participants in total and the best score is 0.81063. The final result of this competition may vary as the score was calculated only with 30% of the test data.

# IV. Conclusion

Thoroughly, we have provided a reasonable problem-solution-fit. During the analysis, we have to constantly make decisions about selection, either selection of the data or the methods to process the data; since the dataset is quite large and messy, this process is not always straightforward and easy. In fact, there is many unclarities of the raw data such as the description of features; ideally, we should had been able to communicate with the dataset provider.

For customer segmentation, Non-negative Matrix Factorization could be an alternative to PCA to extract meaningful features. In the supervised modeling part, by applying advanced hyperparameter tuning technique like Bayesian Optimization, the performance of models could be further improved.

# References

[1] https://scikit-learn.org/stable/auto_examples/plot_roc_curve_visualization_api.html

[2] https://www.graphpad.com/support/faq/what-is-the-difference-between-ordinal-interval-and-ratio-variables-why-should-i-care/

[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[5] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html

[7] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[8] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[9] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[10] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingClassifier.html#sklearn.ensemble.HistGradientBoostingClassifier

[11] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html#sklearn.feature_selection.RFECV

[12] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#examples-using-sklearn-model-selection-gridsearchcv

[13] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html#sklearn.model_selection.RandomizedSearchCV