Project Report

Introduction

What is Customer Segmentation? And Why do we use it?

Customer segmentation is the process of dividing customers into groups based on similar features so that companies can target each group effectively and efficiently. Customers are segmented based on demographic traits such as age, race, religion, income, lifestyle, spending habits, behavioural habits *(Tan et al 2005, Krishna et al. 2013)*

Segmenting customers allows maximum utilization of marketing resources and also helps in staying ahead of the competition by identifying products which existing or potential customers could be interested in.

Customer segmentation is also helpful in determining traits of customers to understand their needs in a better way, as a similar group of people might be interested in the same product or service *(Kotler et al. 2005)*

Customer segmentation is a classification problem and we use supervised and unsupervised machine learning model to predict the correct classes.

Problem Statement: A mail-order company wants to increase its customer base by running a targeted campaign towards individuals who are more likely to become customers. We have the demographic data of the population of Germany and a dataset containing the customers of the mail-order company. Our task is to find the segments of people who are most similar to the current customers. We used use PCA for dimensionality reduction and KMeans clustering algorithm to divide customers into segments *(Kansal et al. 2018)*. We also propose a supervised machine learning framework using AdaBoost as a classifier and a XGBoost model is used as a benchmark.

Algorithms and Techniques: We performed customer segmentation using both Supervised and Unsupervised modes of Machine learning. For unsupervised classification we used PCA for dimensionality reduction because clustering algorithms does not perform well with large dimensions. After performing PCA, data was clustered using Kmeans Clustering algorithm.

For Supervised mode of classification, we trained an AdaBoost (Adaptive Boosting) classifier, which combines multiple weak classifiers into one strong classifier by paying more attention to underfitted training instances by the previous classifier and it also allows use of different classification algorithm as weak learners. Adaboost changes the sample distribution modifying the weights associated with the learners, it increases the weight of incorrectly classified instances and decreases the weight of correctly classified instances at each iteration.

And we compare it against a strong benchmark model XGBoost (Xtreme Gradient Boosting) which is a decision-tree based ensemble Machine learning algorithm that uses gradient boosting framework. XGBoost doesn't modify the sample distribution, the weak learners train on remaining errors called pseudo-residuals of the strong learners. And XGBoost uses gradient descent optimization. The shortcomings of existing weak learners are identified by gradients.

For the Adaboost and XGBoost classifiers, we used sklearn's implementation of Adaboost and scikit-learn wrapper interface for XGBoost.

Datasets and Inputs: We have the demographic data of the population of Germany (*"azidas.csv")*, and the customers of the mail-order company (*"customers.csv"),* which contains 366 features corresponding to the *age, family type, income, social status, neighbourhood,* and several other features. We also have a '*train'* and '*test'* dataset with a label column, which states whether the customer responded to the campaign or not. We will use this '*train'* and '*test'* datasets for creating a supervised Machine learning model.

Description of Dataset:
• ***azdias.csv***: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns).
• ***customers.csv:*** Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns)
• ***Udacity_MAILOUT_052018_TRAIN.csv*****:** Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).
• ***Udacity_MAILOUT_052018_TEST.csv*****:** Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns)
• ***DIAS Information Levels - Attributes 2017:*** Dataframe containing the description of features and the types of values.

Benchmark Model: We use an XGBoost model as a benchmark model, which takes input features as top 'N' principal components obtained after performing Principal Component Analysis (PCA) on the dataset. The model is evaluated using the ROC_AUC score.

Evaluation Metric: Due to the high-class imbalance problem in the dataset, Accuracy would not be a good indicator of the model's performance, so we select Area Under the ROC curve or AUC, which works well in case of an imbalanced dataset.

Following steps were taken in developing the solution:

1. Data Exploration

We have two datasets here, and the first one is the demographic data of the general population of Germany, and the other is the demographic data of a mail-order company.

```
# Loading data
azdias = pd.read_csv('../../data/Term2/capstone/arvato_data/Udacity_AZDIAS_052018.csv', sep=';')
customers = pd.read_csv('../../data/Term2/capstone/arvato_data/Udacity_CUSTOMERS_052018.csv', sep=';')
```

Azdias : (891221, 366), Customers: (191652, 369)

The datasets contain information about the *social* and *financial status* of a person, *the neighbourhood that person lives* is, i.e., *How many families live in that neighbourhood? What type of cars do people drive in the locality? Is the person a money saver or spender?* And many other features.

The dataset is noisy and contains a lot of missing values, some of which are encoded as NaN (fortunately) and others as -1, 9, XX, etc. Each categorical column has a different meaning associated with the values, so we can't just replace all the -1's and 9's with NaN. But for simplicity, we will just consider the NaN as missing values.

2. Data Cleaning

Following steps were taken while cleaning the dataset.

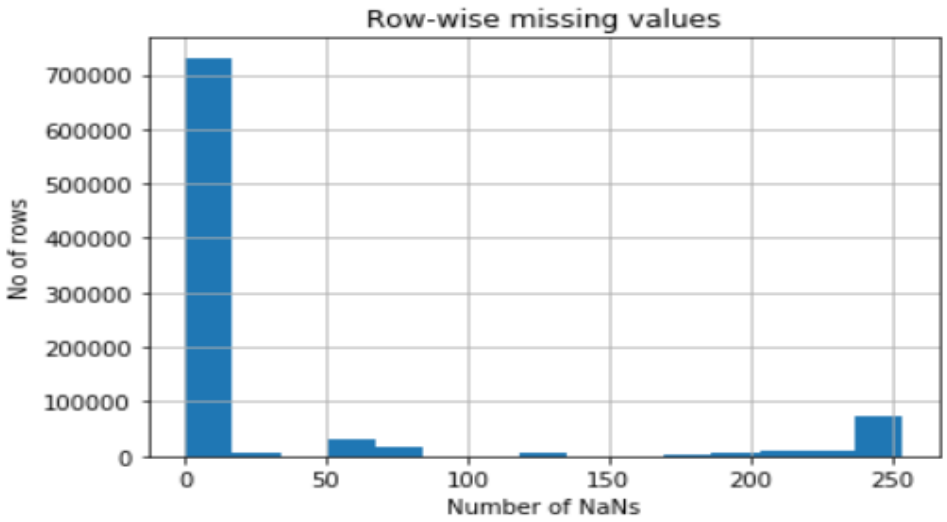
2.1 Finding Missing Values in Columns

Most of the columns contain less than 30% missing values, so those with more than 30% missing values are dropped *(10 columns).*

```
proportion_col = azdias.isnull().sum()/azdias_temp.shape[0]
cols_to_drop = proportion_col[proportion_col > 0.3].index.tolist()
azdias = azdias.drop(cols_to_drop, axis = 1)
```

2.2 Finding Missing values in rows :

We dropped rows containing more than 25 NaN values.



2.3 Dealing with Categorical Columns:

We identified the categorical columns by manually going over the dataset and created dummy variables for them and later the original columns were dropped.

2.4 Imputing Missing Values:

The missing values in the DataFrame were imputed with the mean of the columns.

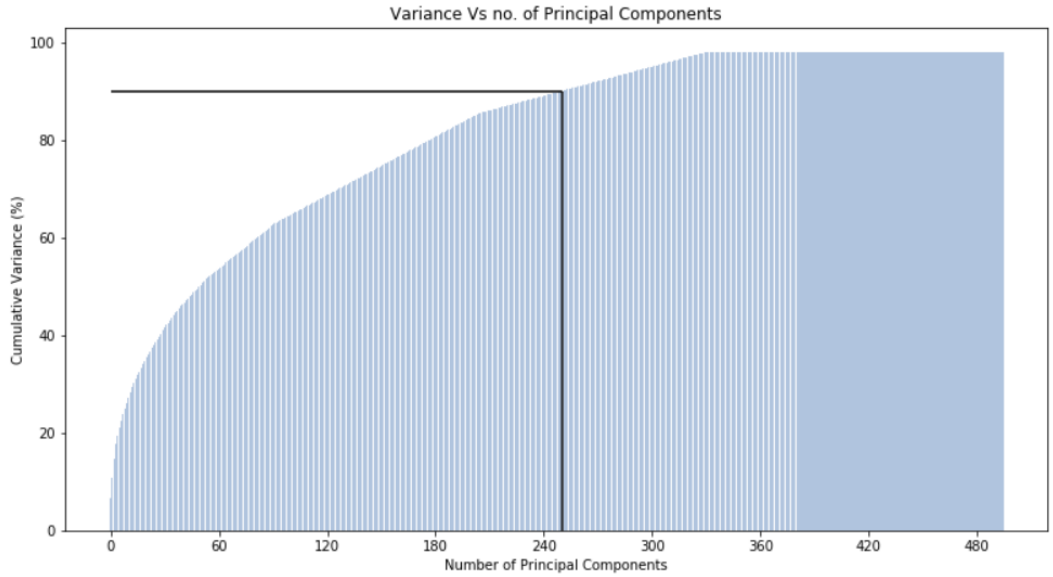
2.5 Scaling Values:

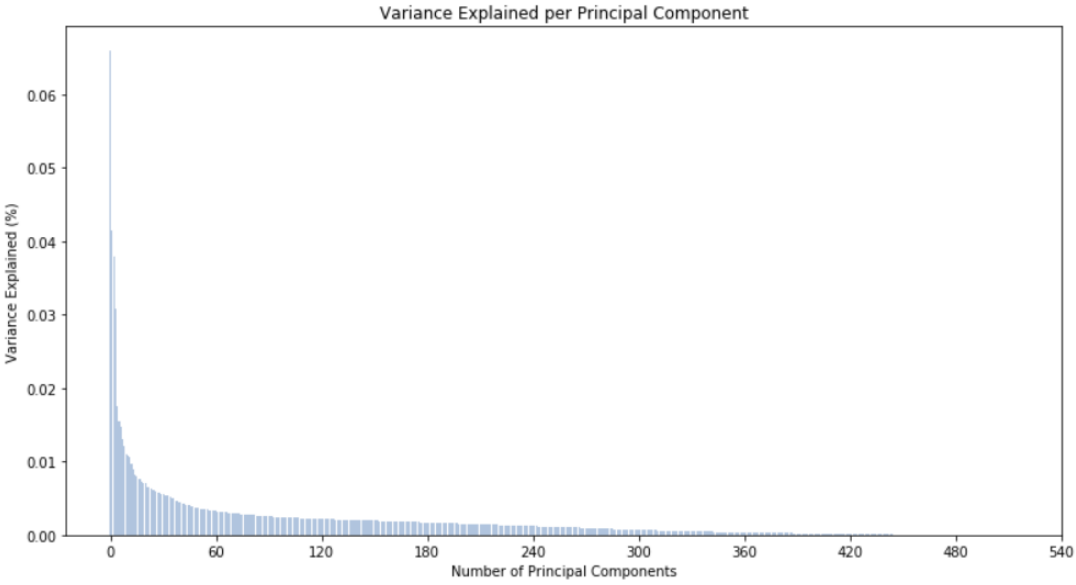
The values in the dataframe takes a range of values which could hurt the performance of algorithms that uses euclidean distance eg KMeans, so the values were scaled in the range -1 to 1 using StandarScaler form sklearn.

3. Principal Component Analysis (PCA)

The dataset originally contains 366 columns, but after the preprocessing and making dummy variables, the number of columns increases to 495. So, we perform PCA on the dataset to reduce the dimensionality and find the principal components which explain the most variability in the dataset.

We perform PCA using the *sklearn*'s implementation of PCA and plot a curve to decide the number of features to keep. It's a common practive to choose number of features that explain 85%-95% of the variance. We selected 250 components, which explain 91%(approx) variance.

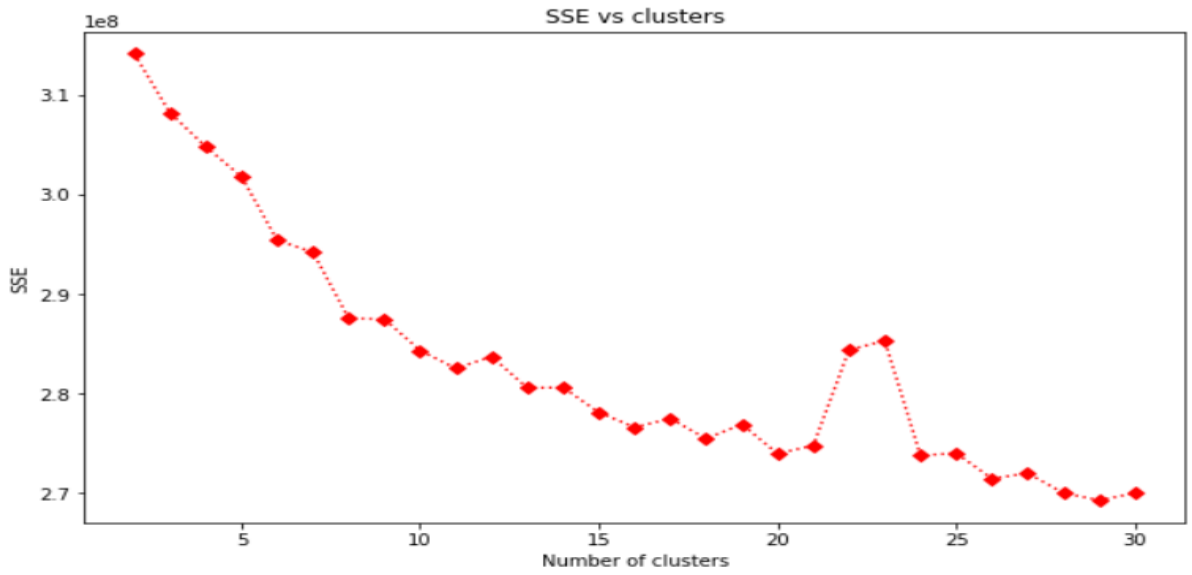




4. Clustering

To segment the customers into different groups or clusters we perform clustering using KMeans algorithm. We perform Mini-Batch KMeans first for a range of clusters

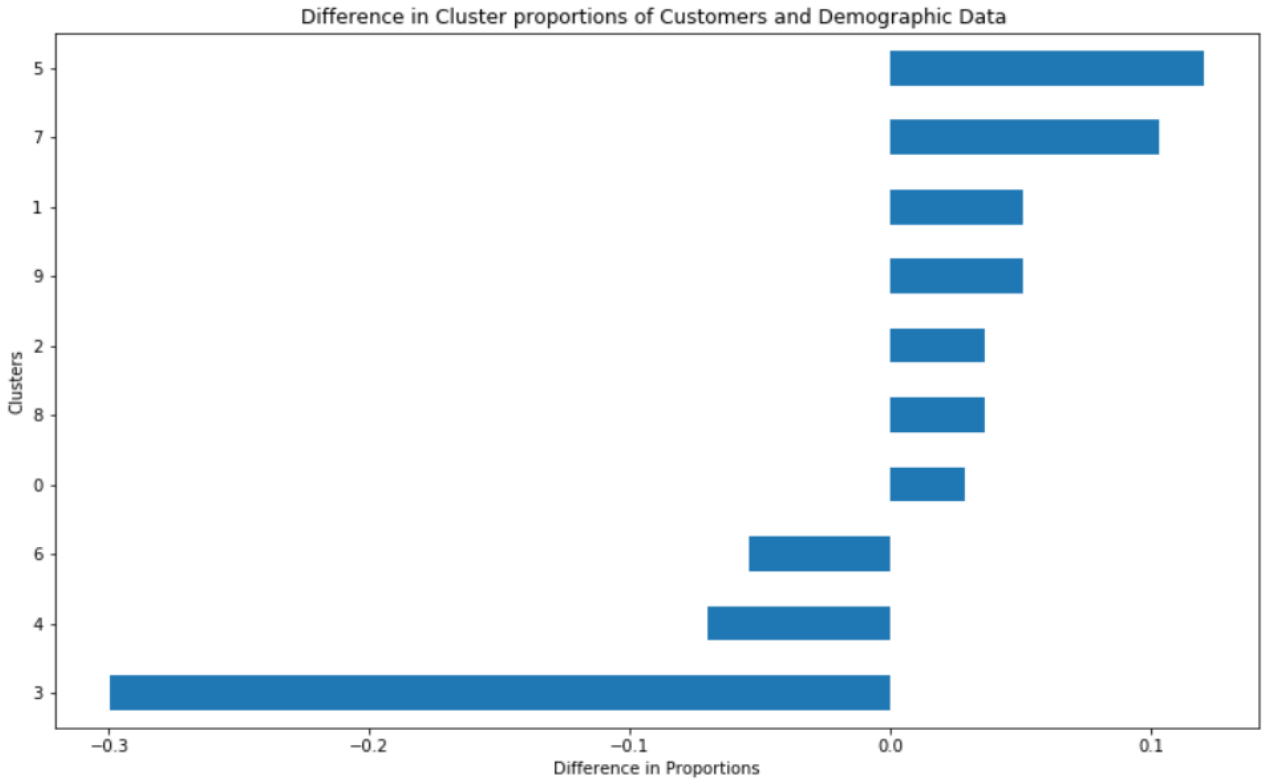
from 2 to 30. And we plot the elbow plot which helps in deciding the optimal number of clusters. The elbow plot is shown below.



There isn't an apparent elbow in the plot. Still, after 10 clusters, we start to get diminishing returns, so we select 10 clusters and refit the data on *azdias* dataset and predict clusters for *customers* dataset.

```
kmeans = KMeans(n_clusters = 10, random_state = 42)
azdias_cluster_labels = kmeans.fit_predict(azdias_pca)
```

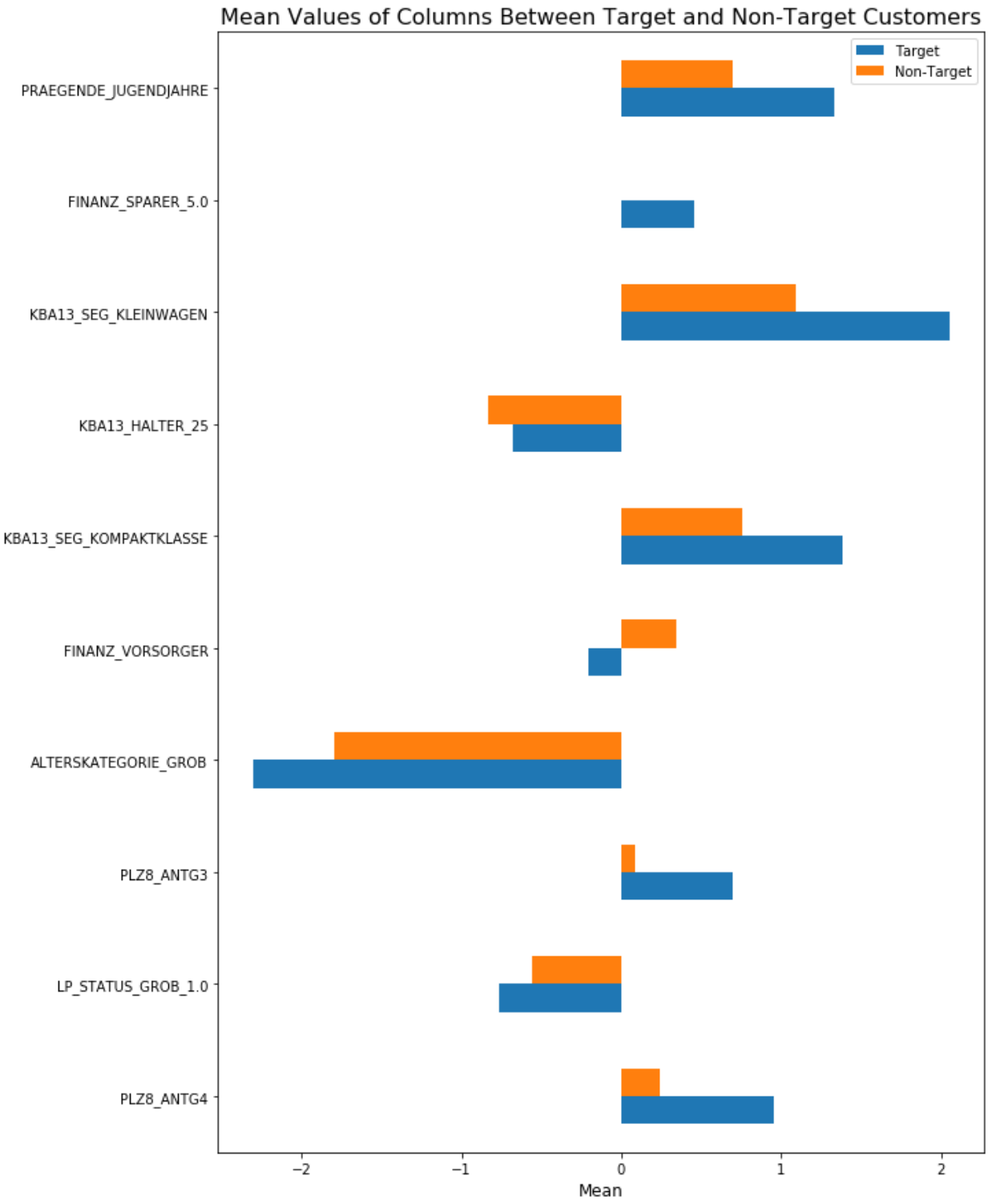
K-Means will return the cluster label for each individual, indicating which cluster a person belongs to. Once we have the labels, we plot a graph showing the difference in the population of the clusters of the *azdias* and *customers* data.



From the above graph, we can see that cluster 5 is an overrepresented cluster (Target audience), and cluster 3 is underrepresented. And cluster 5, 7, contains people who are more similar to the customers of the mail-order company and would respond better to a targeted campaign.

Let's have a closer look at the features of the segmented population by looking at the mean values of the target and non-target customers. We found few features that had very high discriminating power in detecting customers form non-customers.

The plot shows the difference between the columns which helped significantly in segmenting customers (only a few columns are shown in the plot for demonstration purpose).



Feature Description:

- ***PRAEGENDE_JUGENDJAHRE**: dominating movement in the person's youth (avant-garde or mainstream)*

- ***FINANZ_SPARER_5.0****: financial typology: money saver (VERY LOW)*
- ***KBA13_SEG_KLEINWAGEN****: share of small and tiny cars (Ford Fiesta, Ford Ka, etc.) in the PLZ8*
- ***KBA13_HALTER_25****: share of car owners between 21 and 25 within the PLZ8*
- ***KBA13_SEG_KOMPAKTKLASSE****: share of lower midclass cars (Ford Focus etc.) in the PLZ8*
- ***FINANZ_VORSORGER****: financial typology: be prepared*
- ***LP_STATUS_GROB_1.0****: social status (ROUGH)*
- ***ALTERSKATEGORIE_GROB****: age classification through prename analysis*
- ***PLZ8_ANTG3****: number of 6–10 family houses in the PLZ8*
- ***LP_STATUS_GROB_1.0****: social status rough*
- ***PLZ8_ANTG4****: number of >10 family houses in the PLZ8*

Findings:

"The target audience or the people who would respond better to the campaign are between 21 and 25 years of age, with dominating movement in the youth, saves less money and is not financially prepared. They live in a locality where number of family houses is greater than 10 in the PLZ8."

5. Supervised Machine Learning Model

In the unsupervised part, we used clustering to find the people who are more similar to the current customers base or are more likely to become customers of the mail-order company. The features of the overrepresented clusters were used to identify the customers.

Now, in this section, we will look at supervised machine learning models to decide whether a person will become a customer or not.

For this part, we used the MAILOUT dataset which contains information about each individual that was targeted for the mailout campaign and how they responded to it (Response = 0 or 1)

```
mailout_train = pd.read_csv('Udacity_MAILOUT_052018_TRAIN.csv')
mailout_test = pd.read_csv('Udacity_MAILOUT_052018_TEST.csv')
```

The dataset is highly imbalanced and only contains 1.24% data for label 1 (individuals who responded to the campaign), so we will use cross-validation while fitting data to the model and use AUC ROC score as metric to test the model instead of accuracy because of the class imbalance problem.

We train three different models that can handle the class imbalance problem and compare their performance before tuning the final model.

1. *AdaBoost (Proposed Model)*
2. *XGBoost (Benchmark Model)*
3. *GradientBoost (for reference only)*

The AdaBoost model performs the best over the other models with a mean auc_roc score of 79.1846, which is slightly better than the Gradient Boost Model.

The AUC_ROC scores of the models are summarized below:

	AdaBoost	Xgboost	GradientBoost
0	0.734279	0.558609	0.724481
1	0.791846	0.624320	0.788252
2	0.718031	0.590938	0.744663
3	0.773580	0.625504	0.791020
4	0.766238	0.620043	0.771731
5	0.715963	0.620623	0.722465
6	0.768167	0.625265	0.758205

	AdaBoost	Xgboost	GradientBoost
count	7.000000	7.000000	7.000000
mean	0.752586	0.609329	0.757260
std	0.029670	0.025473	0.028150
min	0.715963	0.558609	0.722465
25%	0.726155	0.605490	0.734572
50%	0.766238	0.620623	0.758205
75%	0.770874	0.624792	0.779991
max	0.791846	0.625504	0.791020

Conclusion & Further Improvements

We looked at both Unsupervised and Supervised methods to find the customers who would respond better to a campaign and would probably turn into a customer as a result of a targeted campaign.

The provided solution 'Adaboost' performs significantly better than the benchmark model 'XGBoost'. The 'Adaboost' model gives an increase of 0.1479 in AUC ROC score on an average over 'K' folds.

The model can be improved by optimizing the hyperparameters. Apart from optimization, there are a few things which could lead to better results.

1. A KNN imputer could be used to impute the missing values instead of using mean, median, or mode, which sometimes could add bias to the data.
2. The dataset was analysed manually to look for categorical columns, and not all columns were included because of less clarity about the meaning of the column. A detailed analysis of features could be done to avoid dropping useful features.
3. More models can be trained and stacked together to get better predictions.

References

[1] Kishana R. Kashwan and C. M. Velu, "Customer Segmentation Using Clustering and Data Mining Techniques," *International Journal of Computer Theory and Engineering* vol. 5, no. 6, pp. 856-861, 2013.

[2] T. Kansal, S. Bahuguna, V. Singh and T. Choudhury, "Customer Segmentation using K-means Clustering," 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, Belgaum, India, 2018, pp. 135-139.*

[3] Ph. Kotler and K. Lane Keller. Marketing Management (12th Edition) (Marketing Management). Upper Saddle River: Prentice Hall, 2005.

[4] N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining. (2005) Addison-Wesley, Boston