Customer Segmentation Report and Model for Arvato Financial Solutions

by Luis Camarillo

Table of Contents

Project Definition

Project Overview

The following report presents the capstone project for Udacity's Data Science Nanodegree program, working with data provided by Arvato Financial Solutions, a Bertelsmann subsidiary.

The project is focused with helping a mail-order sales company in Germany interested in identifying segments of the general population to target with their marketing in order to grow. To do this, the project utilizes datasets with demographics from both the general population of Germany as well as customers of the mail-order company. The objective is to use this data to build a model, using both unsupervised and supervised learning techniques, that identifies which individuals are most likely to become customers of the mail-order company.

Problem Statement

The project’s problem statement is relatively simple: How can the mail-order company acquire new clients more efficiently?

To answer this question, I will build a model that identifies which individuals are most likely to become customers of the mail-order company, performing the following tasks:

- Explore and visualize the demographics data available for the general population and existing customers.
- Clean, transform, and prepare the data for modeling.
- Perform unsupervised learning techniques to cluster the data into customer segments.
- Analyze customer segments to describe the relationship between the demographics of the company's existing customers and the general population.
- Build a prediction model with supervised learning techniques targeting customers for a mail-out campaign.
- Iterate and refine the model by trying different algorithms and using cross-validation.
- Evaluate and validate the final model’s performance.

Metrics

Since we are building a binary classifier identifying potential customers better suited for a targeted mail campaign, the model's performance will be evaluated on the area under the curve (AUC) of the Receiver Operating Characteristic (ROC). The ROC curve is a good default for binary classifiers.

The ROC curve plots the true positive rate (TPR, the proportion of actual customers correctly identified) against the false positive rate (FPR, the proportion of non-customers incorrectly labeled as customers).

The AUC, summarizes the performance of the model. A perfect classifier has a ROC AUC score of 1 (all customers are perfectly identified), while a purely random classifier has a ROC AUC score of 0.5. A model that identifies most of the customers first, before starting to make errors, will see its curve start with a steep upward slope towards the upper-left corner before making a shallow slope towards the upper-right.

Analysis

Data Exploration

This project made use of three distinct datasets provided by Arvato Financial Solutions, a Bertelsmann subsidiary, all containing the same basic layout with demographics data intended for a mail-order campaign:

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

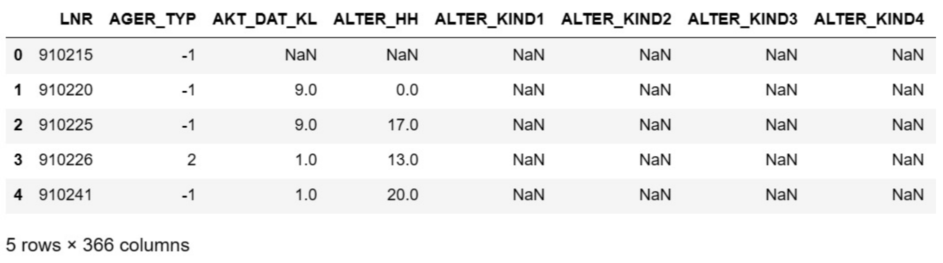
Each row of the datasets represents a single person, but also includes information about the individual's household, building, and neighborhood. The first step of the project will use the first two datasets to figure out how customers ("CUSTOMERS") are similar to or differ from the general population of Germany ("AZDIAS"). The third dataset ("MAILOUT") will be used to predict which individual is most likely to become customer for the mail-order sales company.

The "CUSTOMERS" datasets contains three extra columns ('CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP'), with additional information on the customers on file. In addition, the "MAILOUT" file includes an extra column, "RESPONSE", indicating if the recipient became a customer of the company, to be used as training label.

Outside of these, the remaining columns are shared between the three datasets, retaining the same basic layout. Two Excel spreadsheets were provided in the workspace for additional information on the data: a top-level list of attributes and descriptions, organized by informational category; and detailed mapping of data values for each feature in alphabetical order. These were helpful during initial analysis and visualization.

General Population

The initial analysis for the General Population dataset ("AZDIAS") began by exploring the basic layout, which immediately showed a large number of missing values in the first few rows. The following figure shows an eight of the 366 columns available on this dataset:

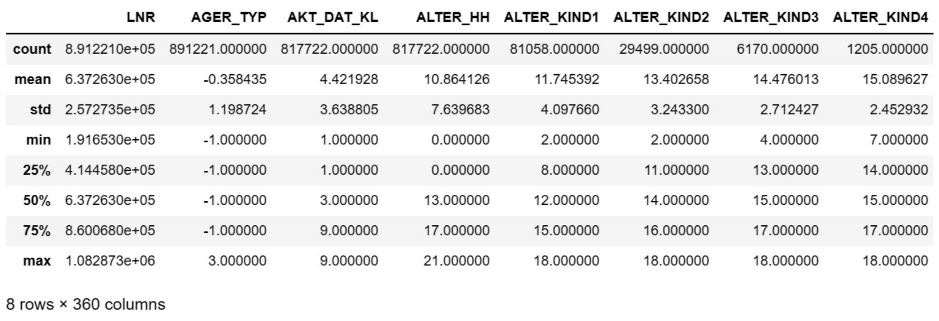


	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
0	910215	-1	NaN	NaN	NaN	NaN	NaN	NaN
1	910220	-1	9.0	0.0	NaN	NaN	NaN	NaN
2	910225	-1	9.0	17.0	NaN	NaN	NaN	NaN
3	910226	2	1.0	13.0	NaN	NaN	NaN	NaN
4	910241	-1	1.0	20.0	NaN	NaN	NaN	NaN

5 rows × 366 columns

Figure 1: General Population dataset ("AZDIAS") with .head()

Next, I applied the .describe() method to the General Population dataset, gaining important insights in terms of the minimum and maximum values, as well as descriptive statistics such as the median and mean per columns. This was particularly interesting when paired with the two excel sheets detailing each attributes description, as it helped identify which columns had values of -1 (like "AGER_TYP" in Figure 1) and 0, often associated with nulls.



	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
count	8.912210e+05	891221.000000	817722.000000	817722.000000	81058.000000	29499.000000	6170.000000	1205.000000
mean	6.372630e+05	-0.358435	4.421928	10.864126	11.745392	13.402658	14.476013	15.089627
std	2.572735e+05	1.198724	3.638805	7.639683	4.097660	3.243300	2.712427	2.452932
min	1.916530e+05	-1.000000	1.000000	0.000000	2.000000	2.000000	4.000000	7.000000
25%	4.144580e+05	-1.000000	1.000000	0.000000	8.000000	11.000000	13.000000	14.000000
50%	6.372630e+05	-1.000000	3.000000	13.000000	12.000000	14.000000	15.000000	15.000000
75%	8.600680e+05	-1.000000	9.000000	17.000000	15.000000	16.000000	17.000000	17.000000
max	1.082873e+06	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000

8 rows × 360 columns

Figure 2: General Population dataset ("AZDIAS") with .describe()

Finally, while applying the .info() method, I was able to identify the null values in the dataset. Several columns, as show in Figure 2 below--like "ALTER_KIN2," "ALTER_KIND3," and "ALTER_KIND4," among others--feature a large proportion of null values. This is without discounting instances with values of -1 and 0, coded as nulls in certain attributes as described by the two Excel spreadsheets provided. Most features are numeric as well, with some (like "CAMEO_DEUG_2015" and "CAMEO_INTL_2015") that must be converted to numeric from wrongly encoded object data type.

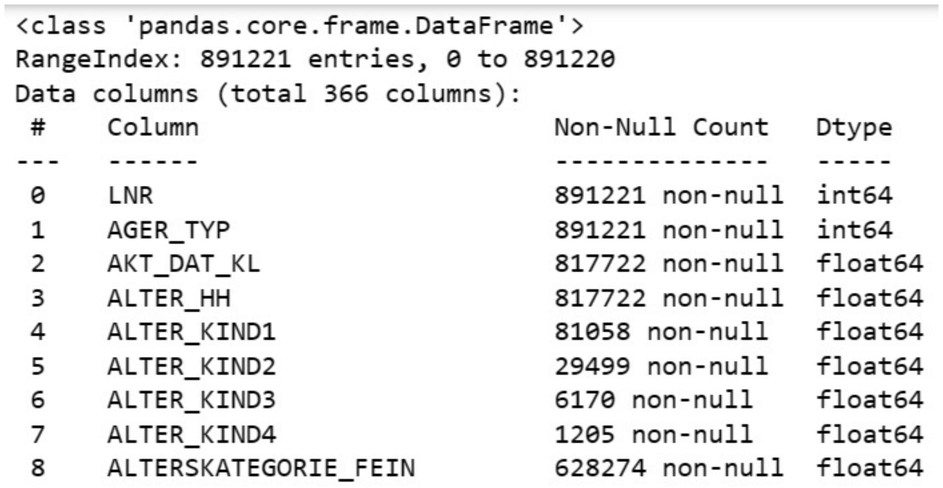
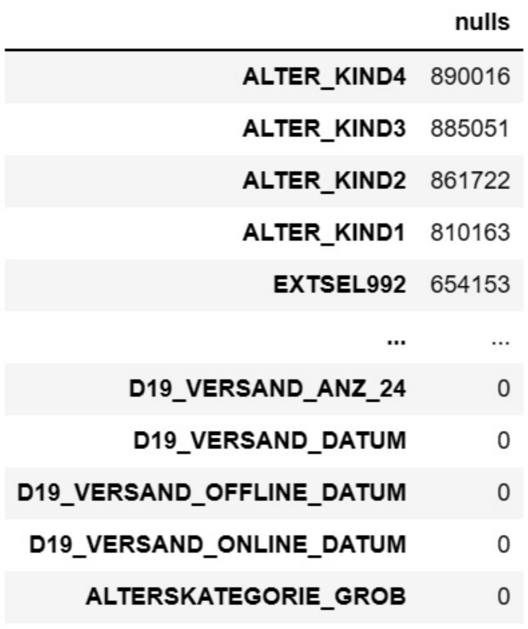

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891221 entries, 0 to 891220
Data columns (total 366 columns):
 #    Column                        Non-Null Count   Dtype  
---   ------                        --------------   -----  
 0    LNR                           891221 non-null  int64  
 1    AGER_TYP                      891221 non-null  int64  
 2    AKT_DAT_KL                    817722 non-null  float64
 3    ALTER_HH                      817722 non-null  float64
 4    ALTER_KIND1                   81058 non-null   float64
 5    ALTER_KIND2                   29499 non-null   float64
 6    ALTER_KIND3                   6170 non-null    float64
 7    ALTER_KIND4                   1205 non-null    float64
 8    ALTERSKATEGORIE_FEIN          628274 non-null  float64
```

Figure 3: General Population dataset ("AZDIAS") with .info()

Now, as shown on the next figure with the top and bottom attributes with null values, not every column features missing values. It would be interesting to investigate further whether it would be appropriate to discard the attributes containing the highest proportion of missing values.



	nulls
ALTER_KIND4	890016
ALTER_KIND3	885051
ALTER_KIND2	861722
ALTER_KIND1	810163
EXTSEL992	654153
...	...
D19_VERSAND_ANZ_24	0
D19_VERSAND_DATUM	0
D19_VERSAND_OFFLINE_DATUM	0
D19_VERSAND_ONLINE_DATUM	0
ALTERSKATEGORIE_GROB	0

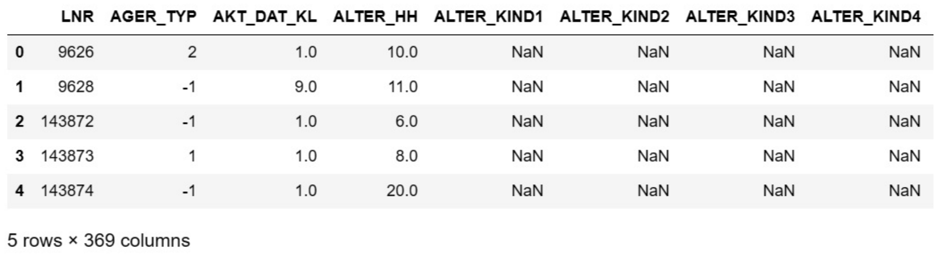
Figure 4: Top and bottom attributes with missing values from General Population dataset ("AZDIAS")

Customers

A similar framework for data exploration was applied for the Customers and

Mailout datasets. The following figures detail the data exploration analysis carried out for the Customers dataset, containing demographics for individuals who are already customers of the company.

The initial analysis for the Customers dataset (CUSTOMERS) showed the same basic layout as the General Population dataset, which allows us to note a large number of missing values in the first few rows for similar columns. The following figure shows an eight of the 369 columns available on this dataset:



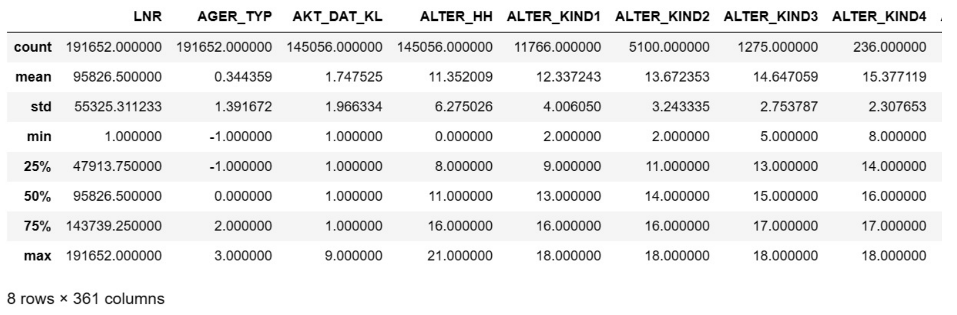
	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
0	9626	2	1.0	10.0	NaN	NaN	NaN	NaN
1	9628	-1	9.0	11.0	NaN	NaN	NaN	NaN
2	143872	-1	1.0	6.0	NaN	NaN	NaN	NaN
3	143873	1	1.0	8.0	NaN	NaN	NaN	NaN
4	143874	-1	1.0	20.0	NaN	NaN	NaN	NaN

5 rows × 369 columns

Figure 5: Customers dataset ("CUSTOMERS") with .head()

As mentioned at the beginning, this dataset contains three additional features specific to customers that are not available for the General Population dataset.

Next, I applied the .describe() method to the Customer dataset, to review descriptive statistics, and identify the same encoded missing values as discussed for the previous dataset. In addition, we are now working with much fewer records, with a count of 191,652 customers compared to the 891,211 people in the General Population dataset.

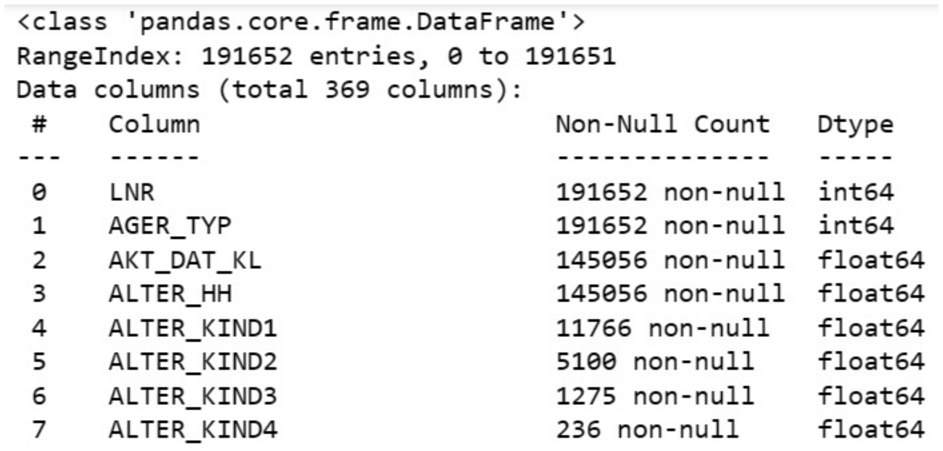


	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
count	191652.000000	191652.000000	145056.000000	145056.000000	11766.000000	5100.000000	1275.000000	236.000000
mean	95826.500000	0.344359	1.747525	11.352009	12.337243	13.672353	14.647059	15.377119
std	55325.311233	1.391672	1.966334	6.275026	4.006050	3.243335	2.753787	2.307653
min	1.000000	-1.000000	1.000000	0.000000	2.000000	2.000000	5.000000	8.000000
25%	47913.750000	-1.000000	1.000000	8.000000	9.000000	11.000000	13.000000	14.000000
50%	95826.500000	0.000000	1.000000	11.000000	13.000000	14.000000	15.000000	16.000000
75%	143739.250000	2.000000	1.000000	16.000000	16.000000	16.000000	17.000000	17.000000
max	191652.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000

8 rows × 361 columns

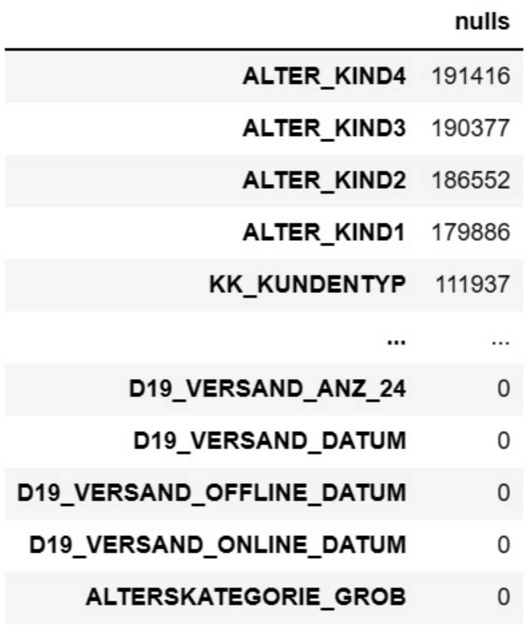
Figure 6: Customers dataset ("CUSTOMERS") with .describe()

Again, the .info() method, allowed me to identify the null values in the dataset, detailing a similar pattern between both datasets in term of missing values and data types.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 191652 entries, 0 to 191651
Data columns (total 369 columns):
 #    Column                         Non-Null Count   Dtype
---   ------                         --------------   -----
 0    LNR                            191652 non-null  int64
 1    AGER_TYP                       191652 non-null  int64
 2    AKT_DAT_KL                     145056 non-null  float64
 3    ALTER_HH                       145056 non-null  float64
 4    ALTER_KIND1                    11766 non-null   float64
 5    ALTER_KIND2                    5100 non-null    float64
 6    ALTER_KIND3                    1275 non-null    float64
 7    ALTER_KIND4                    236 non-null     float64
```

Figure 7: Customers dataset (“CUSTOMERS”) with .info()

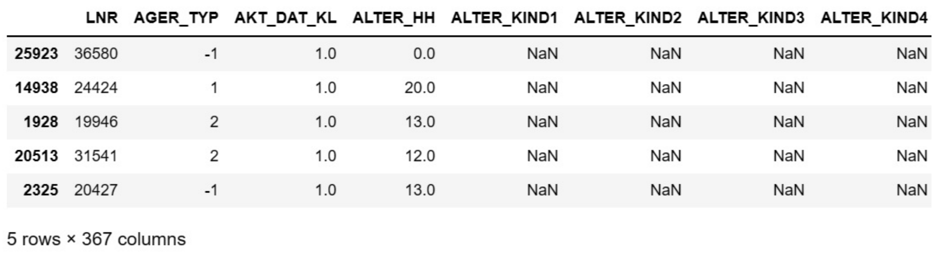


	nulls
ALTER_KIND4	191416
ALTER_KIND3	190377
ALTER_KIND2	186552
ALTER_KIND1	179886
KK_KUNDENTYP	111937
...	...
D19_VERSAND_ANZ_24	0
D19_VERSAND_DATUM	0
D19_VERSAND_OFFLINE_DATUM	0
D19_VERSAND_ONLINE_DATUM	0
ALTERSKATEGORIE_GROB	0

Figure 8: Top and bottom attributes with missing values from Customers dataset (“CUSTOMERS”)

Mailout

For the Mailout dataset (“MAILOUT”) we are dealing same basic layout as the General Population dataset, with an additional “RESPONSE” column indicating whether the person became a customer, which will be used as a label to train and evaluate the eventual model. It is important to note that the Mailout dataset was further divided into training and test sets (at an 80-20 ratio), stratified by the “RESPONSE” variable, to allow for evaluation, and that the exploratory analysis was done exclusively on the training set, as to prevent data snooping bias. The following figure shows eight of the 367 columns available on this dataset, also showing several null values:

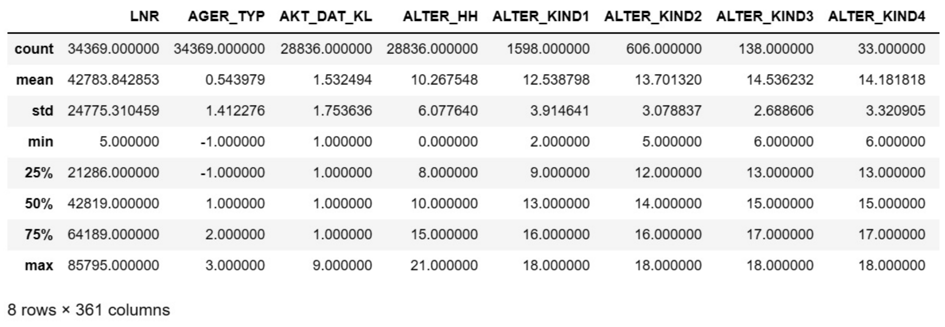


	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
25923	36580	-1	1.0	0.0	NaN	NaN	NaN	NaN
14938	24424	1	1.0	20.0	NaN	NaN	NaN	NaN
1928	19946	2	1.0	13.0	NaN	NaN	NaN	NaN
20513	31541	2	1.0	12.0	NaN	NaN	NaN	NaN
2325	20427	-1	1.0	13.0	NaN	NaN	NaN	NaN

5 rows × 367 columns

Figure 9: Mailout train dataset ("MAILOUT") with .head()

Next, I applied the .describe() method to the Mailout train dataset, to review descriptive statistics, and identify the same encoded missing values as discussed for the previous dataset. Like with the Customers dataset, we are now working with much fewer records: 34,369 individuals, which represent 80% of the original Mailout dataset.

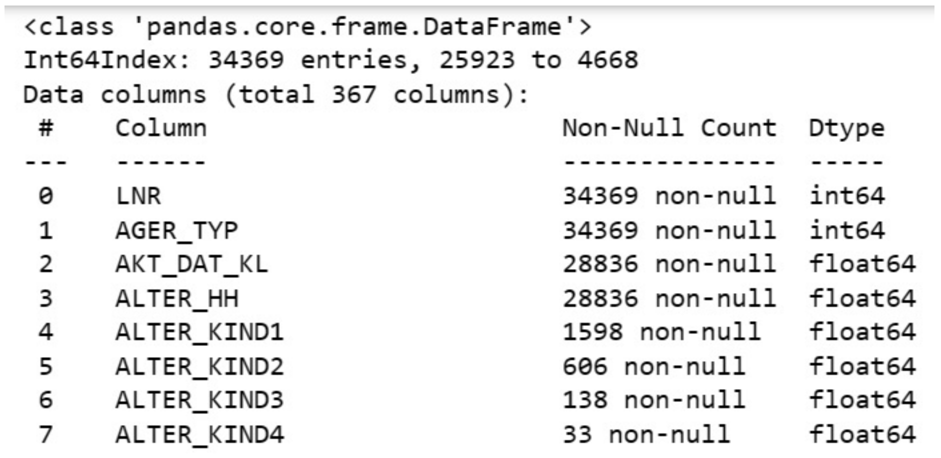


	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4
count	34369.000000	34369.000000	28836.000000	28836.000000	1598.000000	606.000000	138.000000	33.000000
mean	42783.842853	0.543979	1.532494	10.267548	12.538798	13.701320	14.536232	14.181818
std	24775.310459	1.412276	1.753636	6.077640	3.914641	3.078837	2.688606	3.320905
min	5.000000	-1.000000	1.000000	0.000000	2.000000	5.000000	6.000000	6.000000
25%	21286.000000	-1.000000	1.000000	8.000000	9.000000	12.000000	13.000000	13.000000
50%	42819.000000	1.000000	1.000000	10.000000	13.000000	14.000000	15.000000	15.000000
75%	64189.000000	2.000000	1.000000	15.000000	16.000000	16.000000	17.000000	17.000000
max	85795.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000

8 rows × 361 columns

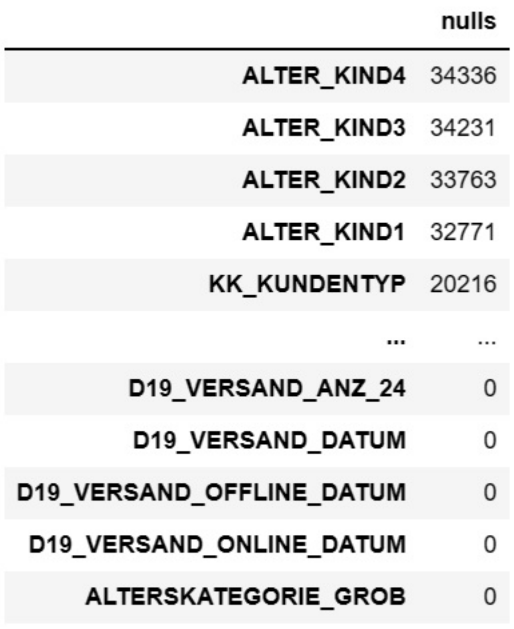
Figure 10: Mailout train dataset ("MAILOUT") with .describe()

Again, the .info() method helps with identifying the null values in the dataset, detailing a similar pattern between the datasets in terms of missing values and data types.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 34369 entries, 25923 to 4668
Data columns (total 367 columns):
 #    Column                      Non-Null Count  Dtype
---   ------                      --------------  -----
 0    LNR                         34369 non-null  int64
 1    AGER_TYP                    34369 non-null  int64
 2    AKT_DAT_KL                  28836 non-null  float64
 3    ALTER_HH                    28836 non-null  float64
 4    ALTER_KIND1                 1598 non-null   float64
 5    ALTER_KIND2                 606 non-null    float64
 6    ALTER_KIND3                 138 non-null    float64
 7    ALTER_KIND4                 33 non-null     float64
```

Figure 11: Mailout train dataset ("MAILOUT") with .info()



	nulls
ALTER_KIND4	34336
ALTER_KIND3	34231
ALTER_KIND2	33763
ALTER_KIND1	32771
KK_KUNDENTYP	20216
...	...
D19_VERSAND_ANZ_24	0
D19_VERSAND_DATUM	0
D19_VERSAND_OFFLINE_DATUM	0
D19_VERSAND_ONLINE_DATUM	0
ALTERSKATEGORIE_GROB	0

Figure 12: Top and bottom attributes with missing values from Mailout train dataset ("MAILOUT")

Data Visualization

General Population

Following the Data Exploration phase, I investigated which features had over 20% of null values, as visualized in the following figure:

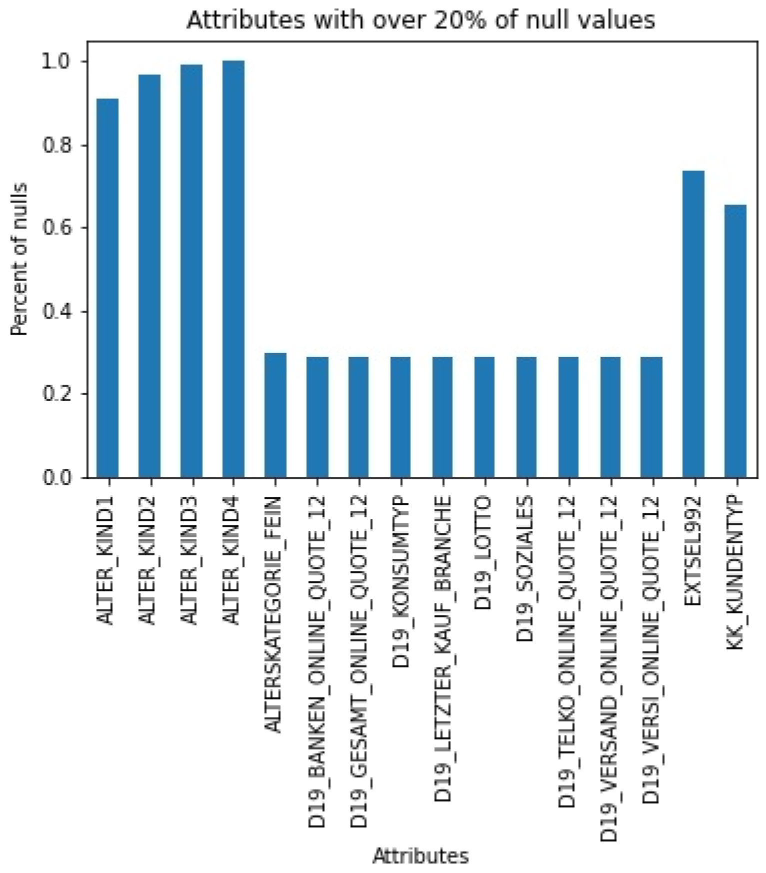


Figure 13: General Population dataset ("AZDIAS") features with over 20% missing values

I also decided to visualize a few key features from the datasets. The General Population dataset features a slightly higher population of Female individuals in comparison to people who identify as Male, consistent with what you would expect of Germany's population.

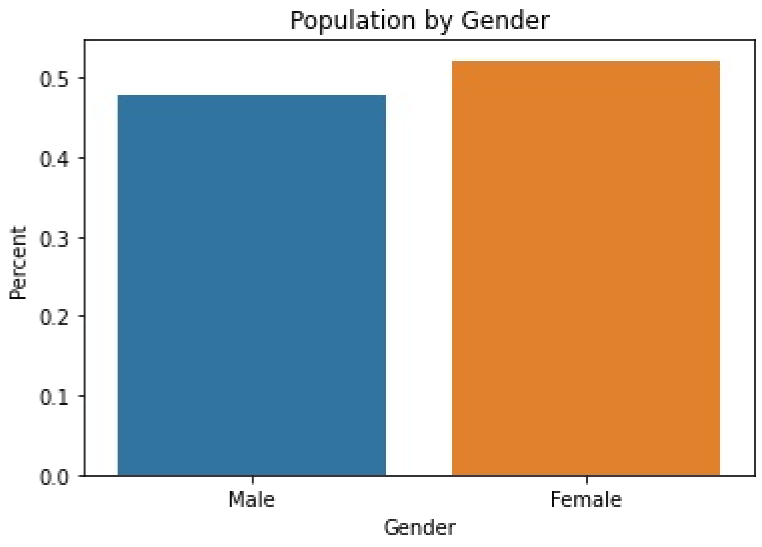


Figure 14: General Population dataset (“AZDIAS”) by Gender

At the same time, in terms of year of birth, the General Population data shows a sharp decline around the year 2000, with most records coming from 1960-1980, probably due to how easy it is to get demographics data from adults.

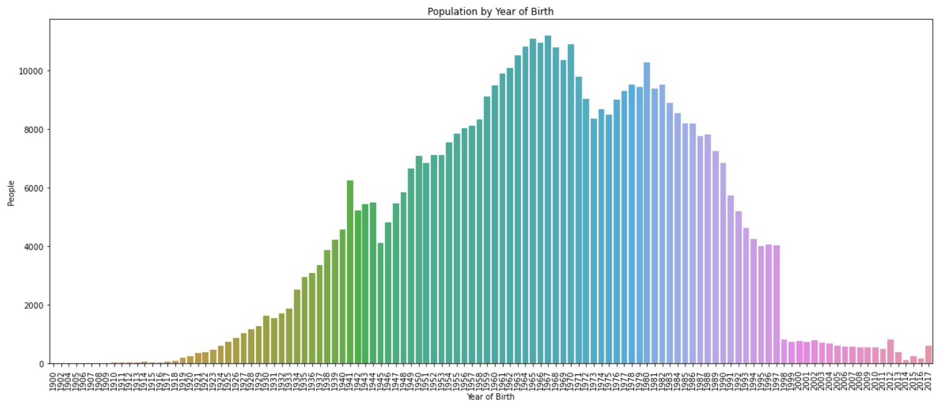


Figure 15: General Population dataset (“AZDIAS”) by Year of Birth

The following histogram shows the distribution of the General Population by Year of Birth:

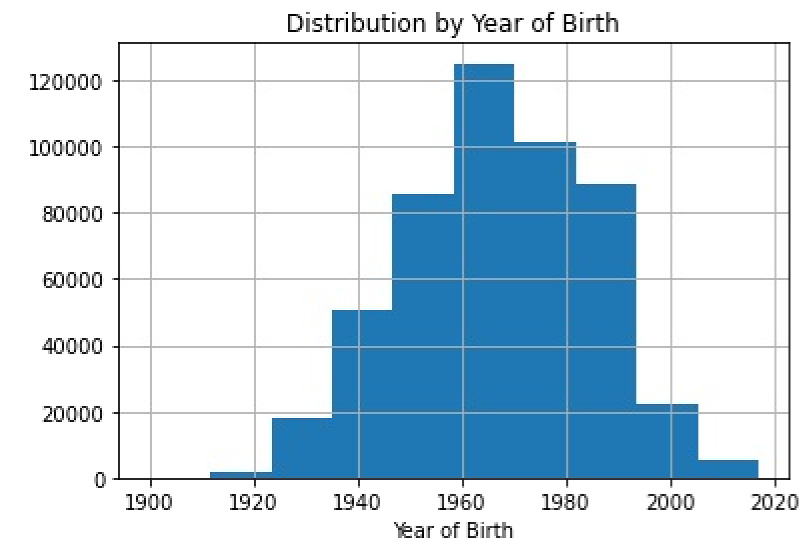


Figure 16: General Population dataset ("AZDIAS") distribution by Year of Birth

Customers

With regards to the Customers dataset, it contains a considerably larger proportion of missing values in each column, so I upped the threshold to 30% to be able to read the attribute names. This graph shows a similar number of features with over 30% of null values, as visualized in the following figure:

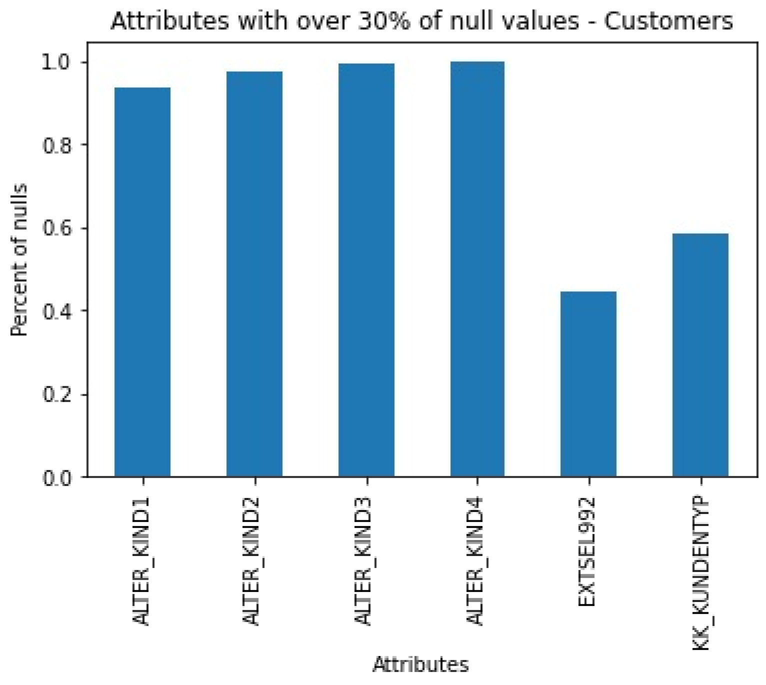


Figure 17: Customers dataset ("CUSTOMERS") features with over 30% missing

values

However, in terms of gender balance, the Customers dataset is skewed towards males, differing considerably from Germany’s overall population, shown in the previous section.

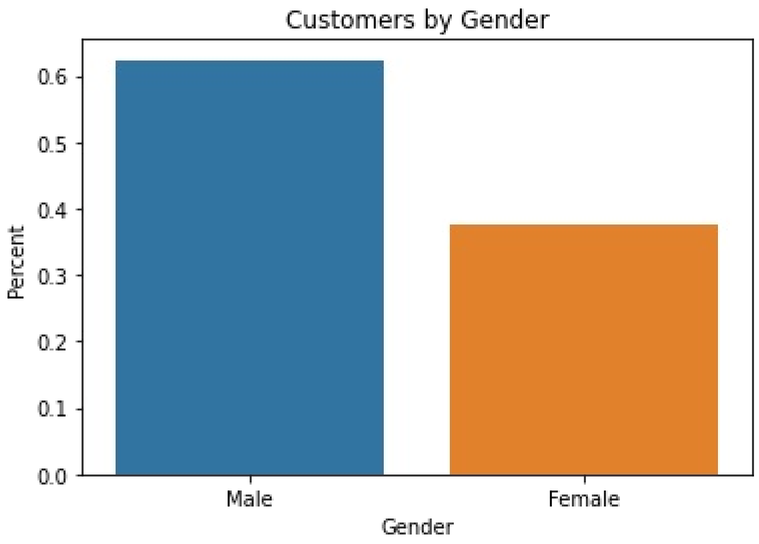


Figure 18: Customer dataset (“CUSTOMERS”) by Gender

Again, in terms of year of birth, the Customers dataset differs from the General Population, skewing towards a much older population. This, couple with the previous graph, suggests that older men, possibly with higher disposable income, are the main customers of the mail-order company -- something that we will be able to discuss after some deeper analysis.

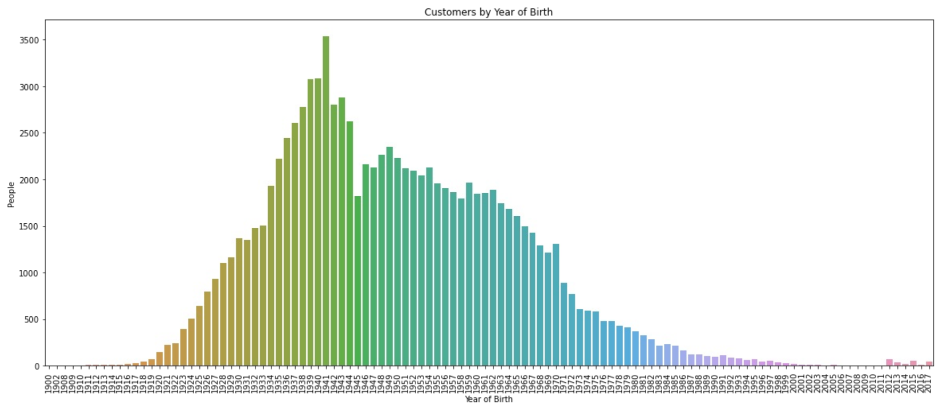


Figure 19: Customers dataset (“CUSTOMERS”) by Year of Birth

The following histogram shows the distribution of the Customers by Year of Birth, clearly right-skewed due to older customers:

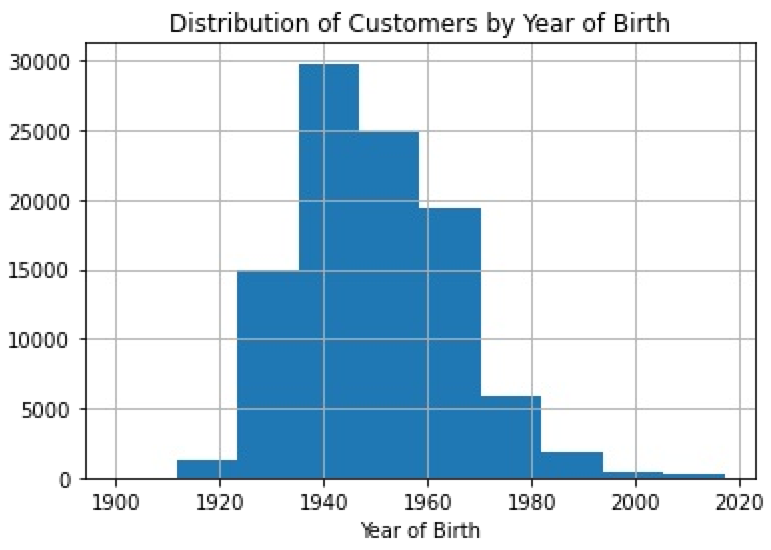


Figure 20: Customers dataset (“CUSTOMERS”) distribution by Year of Birth

General Population vs. Customers

I created further visualizations to compare both datasets on these key features. The following figure compares gender between the Customers dataset and the General population, emphasizing the already discussed gender imbalance.

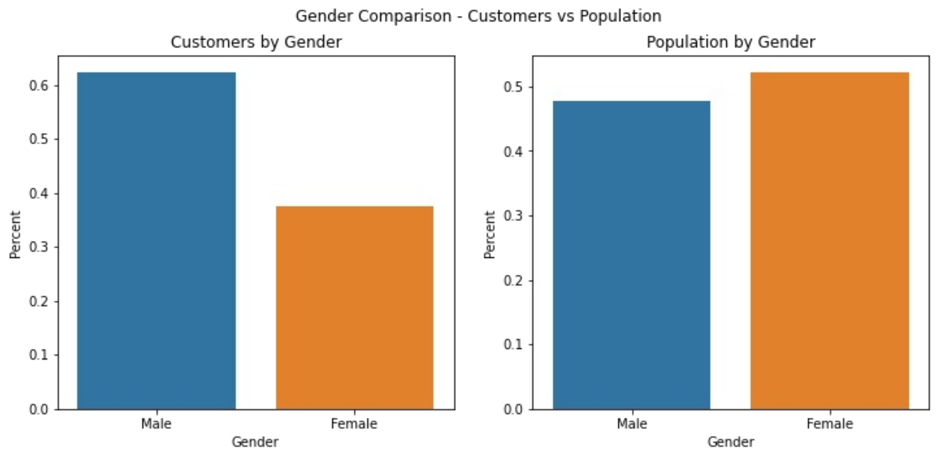


Figure 21: Customers and General Population Gender Comparison

Next, I overlapped the histograms of the birth year for each of the datasets, with the General Customers dominating the visualization due to its larger size. However, it still serves to illustrate the difference in year of birth mode.

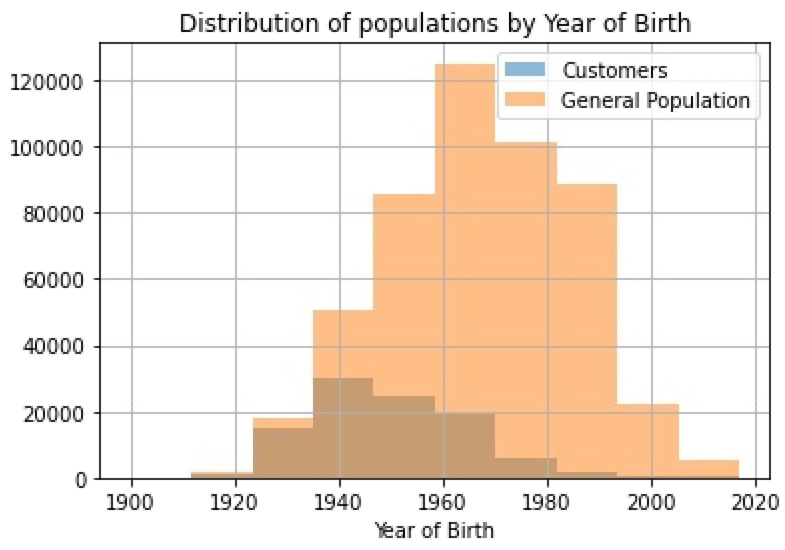


Figure 22: Customers and General Population distribution by Year of Birth

The comparison can be shown more clearly with the following density plot, which emphasizes that customers peak at a much older population.

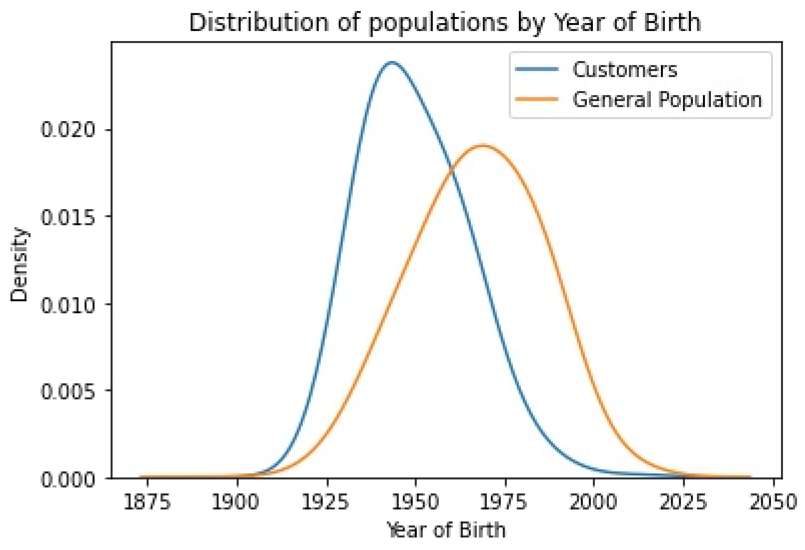


Figure 23: Customers and General Population distribution density plot by Year of Birth

Mailout

Finally, similar visualizations were created for the Mailout dataset. With regards to missing values, it shows a similar number of features with over 20% of null values, as visualized in the following figure:

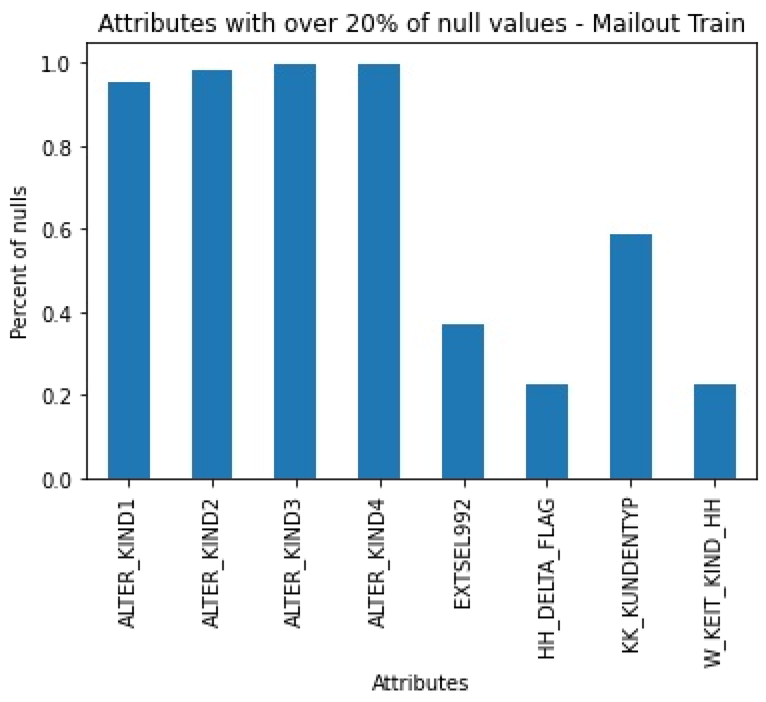


Figure 24: Mailout dataset ("MAILOUT") features with over 20% missing values

Curiously, in terms of gender balance, the Mailout dataset is skewed considerably towards the female population, much more than the General Population dataset an in seemingly direct opposition to the previously analyzed Customers dataset.

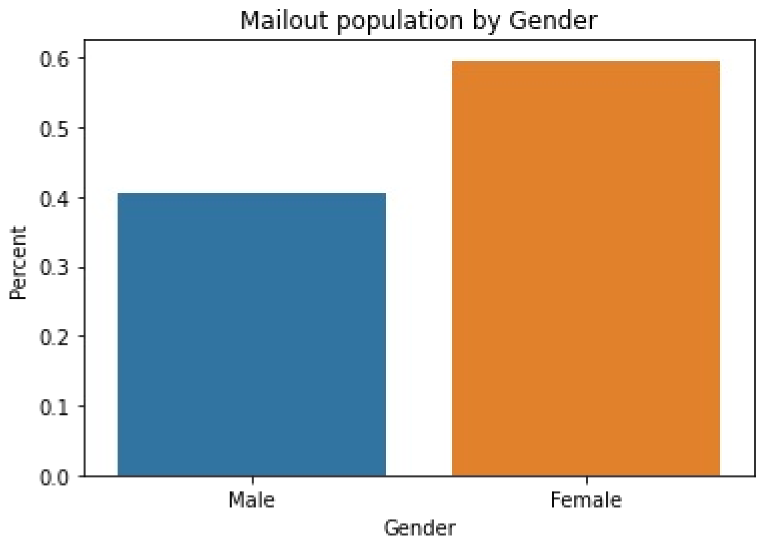


Figure 25: Mailout training dataset ("MAILOUT") by Gender

Again, in terms of year of birth, the Mailout dataset skews older, in part, I imagine, as it would be a terrible idea to target children for a mail-order campaign.

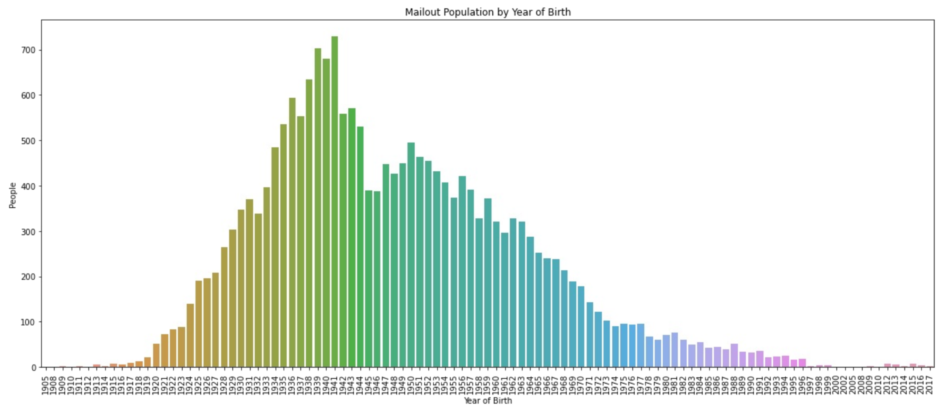


Figure 26: Mailout training dataset ("MAILOUT") by Year of Birth

The following histogram shows the distribution of the Mailout recipients by Year of Birth, clearly right-skewed due to older customers, very similar to the Customers dataset's distribution:

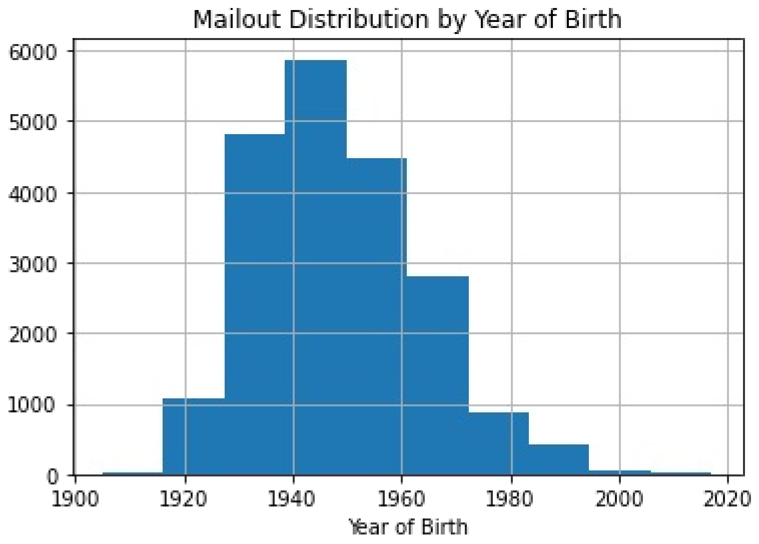


Figure 27: Mailout training dataset ("MAILOUT") distribution by Year of Birth

And finally, I wanted to see how imbalanced the dataset was in regards to the "RESPONSE" attribute, assuming (correctly, it turns out) that relatively few people targeted in the Mailout set actually became customers: just over 1.2%.

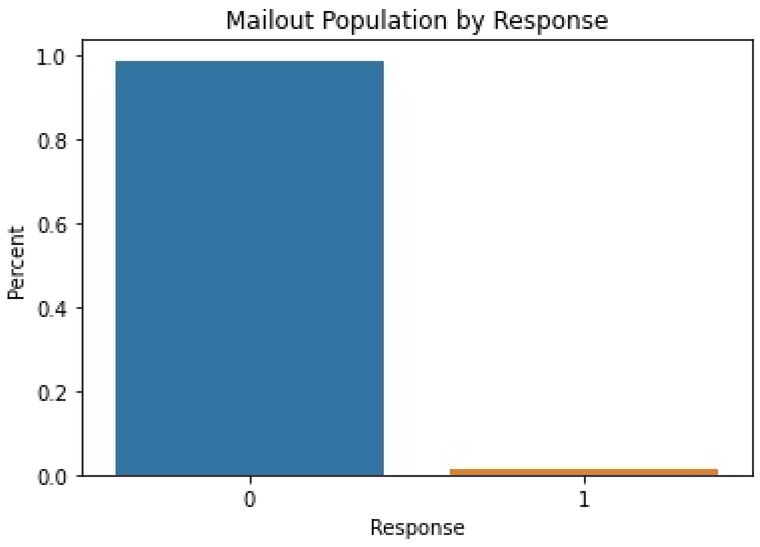


Figure 28: Mailout training dataset (“MAILOUT”) population by response

Methodology

Data Preprocessing

The following data processing steps were applied to all three datasets--General Population, Customers, and Mailout--as all were presented in the same general layout and contained similar errors and required cleaning.

The steps are as follows:

- The code pulls in the IAS Attributes - Values 2017 Excel spreadsheet, which indicates which values (usually -1, occasionally 0) are being used to encode missing values outside of the traditional nulls we already saw in the Analysis section.
- These values are mapped to their corresponding attribute for easy access.
- The Customers and Mailout datasets drop the extra columns not present in the Population dataset, as these are not useful for general analysis.
- In a dedicated Data Preparation function:
    - Missing values encoded as -1 or 0 are converted to nulls (using the aforementioned IAS Attributes - Values 2017 Excel spreadsheet as reference).
    - Columns with over 20% missing values are dropped -- previously identified in the Analysis section -- ignoring attributes that don't add much value.
    - Rows with more than 10 missing values are also dropped, discarding records without much overall value.
    - Numeric columns with object data types (previously identified during the Data Exploration section) are converted to numeric.
    - Missing values are imputed through the attribute's mode.
    - Categorical values are encoded using One Hot Encoding.
- Afterwards, scaling is applied through standardization, which will be helpful during Principal Component Analysis applied during the Implementation phase.

After applying the previously described data preparation steps, null values and attributes are either completely discarded or imputed, resulting in clean datasets without missing values, correctly encoded, and scaled, ready for modeling.

Implementation

Customer Segmentation Report

After preparing the two datasets, General Population and Customers, through the data preparation and cleaning function, and scaling the values, I proceeded to implement Principal Component Analysis with the intent of reducing the dimensionality of the datasets prior to implementing the segmentation algorithm. I was able to reduce the number of features from 366 to 40, therefore reducing processing time in the next steps.

With much smaller arrays, the next step involved applying a K-Means clustering algorithm. K-means allows you to cluster the datasets on any given number of predefined clusters, in this case 20. After applying K-Means to both datasets, I ended up with two data frames of individuals, one for Customers and one for the General Population, each labeled with their corresponding cluster.

Supervised Learning Model

After splitting the Mailout dataset into train and test sets and performing the data preparation, cleaning, and scaling preprocessing steps, I was tested several classification models to pick the best performers.

First, it is important to recognize that we are working with highly imbalanced data--as explored in the Data Visualization section, the new customers gained from the Mailout dataset (as indicated by the “RESPONSE” variable) account for just over 12% of the overall data.

To make it easier on our models, I compensated using an oversampling technique called Adaptive Synthetic Sampling, or ADASYN from the imbalanced-learn package, which is `similar method to SMOTE, but generates different number of samples depending on an estimate of the local distribution of the class to be oversampled`

I ended up creating a pipeline that first applies ADASYN, oversampling the data before fitting a model, therefore working towards minimizing the effects of the imbalanced dataset.

Next, I ran a quick for loop over 10 classification models, applying the previously describe over- and undersampling pipeline and fitting each of the classification models. Afterwards, using 5-fold cross validation scoring on ROC AUC to reduce the risk of overfitting, I was able to evaluate the top performers.

The models tested include:

- XGBClassifier
- GradientBoostingClassifier
- DecisionTreeClassifier
- RandomForestClassifier
- LinearSVC
- SVC
- LogisticRegression
- KNeighborsClassifier
- SGDClassifier
- AdaBoostClassifier

The best performing classification models with the Extreme Gradient Boosting Classifier (XGBClassifier), with ROC AUC scores of 0.632.

The next section, Refinement, will focus on hyperparameter tuning this model.

Refinement

Customer Segmentation Report

For the Principal Component Analysis, reducing the number of features down to 40 was chosen after applying PCA on the General Population dataset with 100 components, and creating a Scree Plot, the visual representation of eigenvalues that define the magnitude of eigenvectors, also known as the principal components. Using the explained variance from scikit-learn's PCA and plotting it below, we can see that after 40 features, the values remain steady, so there is no need to keep more of them.

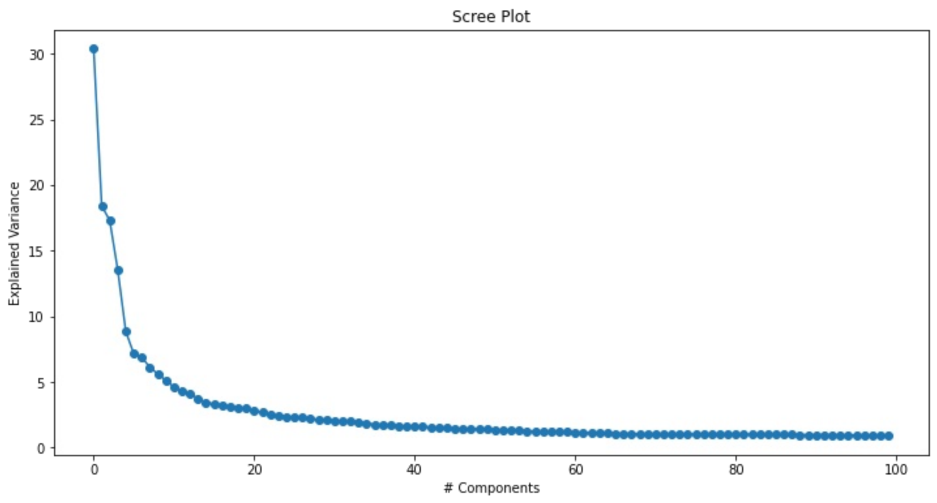


Figure 29: PCA Scree Plot for General Population Dataset

To pick the number of clusters fro the K-Mean analysis, a similar method was implemented. Using a for loop to iterate over a cluster size of up to 30 clusters, I was able to create several K-Means models and saving their inertia. With this information, we can create a chart that shows that beyond 20 clusters, the inertia value barely changes, indicating the ideal value of clusters.

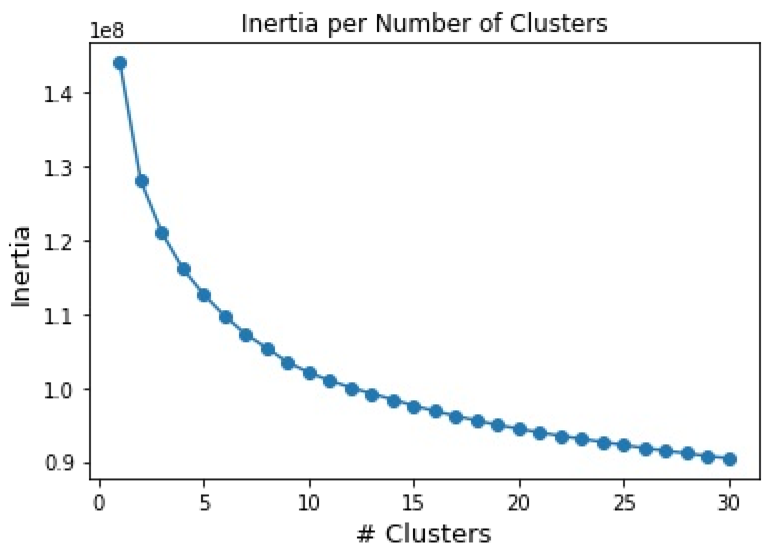


Figure 30: K-Means Inertia Plot

Supervised Learning Model

After picking best performing classification model, Extreme Gradient Boosting Classifier (XGBClassifier) with a crossvalidated ROC AUC scores of 0.632, I implemented a set of grid searches with cross validation at three folds.

First, I created a grid to tune the following hyperparameters with standard starting values, running 972 fits:

- "model__max_depth": [3, 4, 5, 7]
- "model__learning_rate": [0.1, 0.01, 0.05]
- "model__gamma": [0, 0.25, 1]
- "model__reg_lambda": [0, 1, 10]
- "model__scale_pos_weight": [1, 3, 5]
- "model__subsample": [0.8]
- "model__colsample_bytree": [0.5]

This resulted in a cross validated ROC AUC score of 0.6789, 0.0468 better than the untuned model, and the following optimized hyperparameters:

- 'model__colsample_bytree': 0.5
- 'model__gamma': 1
- 'model__learning_rate': 0.01
- 'model__max_depth': 4
- 'model__reg_lambda': 1
- 'model__scale_pos_weight': 1
- 'model__subsample': 0.8

After, I implemented a second run trying to gain a better model, keeping the hyperparameters with values in the middle of the original grid (like learning_rate and max_depth), and considering further tuning for more extreme values. The following grid was proposed:

- "model__gamma": [1, 3, 5, 7]
- "model__subsample": [0.5, 0.7, 0.9]
- "model__colsample_bytree": [0.5, 0.7, 0.9]

This second iteration resulted in a cross validated ROC AUC score of 0.6806, a slight improvement to the previous model, and the following optimized hyperparameters:

- 'model__colsample_bytree': 0.9
- 'model__gamma': 5
- 'model__learning_rate': 0.01
- 'model__max_depth': 4
- 'model__reg_lambda': 1
- 'model__scale_pos_weight': 1
- 'model__subsample': 0.9

Lastly, I proposed I final run, keeping the center values with the following hyperparameter grid for further tuning:

- "model__gamma": [4, 5, 6]
- "model__subsample": [0.8, 0.9, 1]
- "model__colsample_bytree": [0.8, 0.9, 1]

This run resulted in a ROC AUC score of 0.6803, a slight underperformance when compared to the last model, so I decided to stop tuning here and use the second tuned iteration as the final model. This had a final cross validated ROC AUC score of 0.6806, an improvement of 0.05 over the untuned original model. The final model has the following characteristics:

- 'model__colsample_bytree': 0.9
- 'model__gamma': 5
- 'model__learning_rate': 0.01
- 'model__max_depth': 4
- 'model__reg_lambda': 1
- 'model__scale_pos_weight': 1
- 'model__subsample': 0.9

Results

Model Evaluation and Validation

Customer Segmentation Report

After applying the clustering model to both the General Population and Customers datasets and assigning labels to each person recorded, we can compare the population of each cluster.

As the following figure shows, the General population has more uniform clusters, which makes sense, considering that the clustering algorithm was fitted a dataset that is presumably representative of the overall population of Germany. Customers, however, will probably be members of very specific clusters that share the characteristics of someone interested in the products the company sells. This can also be seen clearly in the next figure, as some clusters are considerably overrepresented within the customer population, while others are barely present.

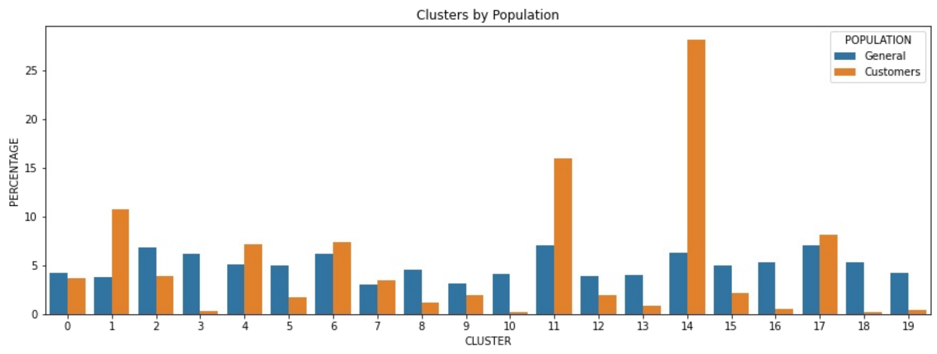
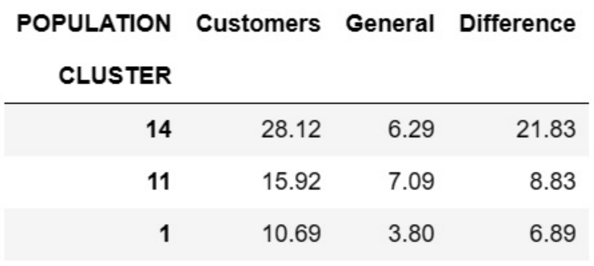


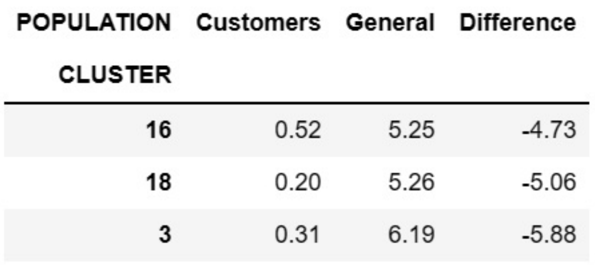
Figure 31: Clusters by percentage of population for the General Population and Customers datasets

In the following two figures, you can see the top three segments overrepresented within the customer base, and the bottom three underrepresented, compared with the overall population of Germany, alongside the difference in percentage.



POPULATION CLUSTER	Customers	General	Difference
14	28.12	6.29	21.83
11	15.92	7.09	8.83
1	10.69	3.80	6.89

Figure 32: Top three overrepresented customers segments



POPULATION CLUSTER	Customers	General	Difference
16	0.52	5.25	-4.73
18	0.20	5.26	-5.06
3	0.31	6.19	-5.88

Figure 33: Top three underrepresented customers segments

In the next section, we will explore more thoroughly the two clusters more overrepresented and underrepresented with customers.

Supervised Learning Model

The final model is an Extreme Gradient Boosting classifier with fitted data run through an ADASYN oversampling technique to combat the imbalanced nature of the training data. It features the following hyperparameter characteristics:

- `'model__colsample_bytree': 0.9`
- `'model__gamma': 5`
- `'model__learning_rate': 0.01`
- `'model__max_depth': 4`
- `'model__reg_lambda': 1`
- `'model__scale_pos_weight': 1`
- `'model__subsample': 0.9`

The final model was evaluated on the test data created through a 20% stratified test-train split. This data was not used during the initial analysis and modeling, but was preprocessed and scaled in the same way as the training data.

After fitting the final model to the training data, and predicting on the test data and comparing with the test labels, I used scikitlearn functions to calculate and Area under ROC curve of 0.74, which is much better than a purely random classifier (0.5 ROC AUC score) but could have room for improvement.

Out of the 6,604 individuals in the test set, the final model identified 750 as people the mailorder company could target and turn into customers--roughly 11% of the test population.

Justification

Customer Segmentation Report

As discussed in the previous section, after applying the clustering algorithm to the General Population and Customers datasets, 20 customers segments were able to be derived from the data. This section will discuss two customers segments that differ considerably from the general population, indicating segments the company might want to focus on (as it makes up for a plurality of its customers) or not prioritize (as they have very few customers with this characteristics).

With the following figures, I aim to explore Cluster 14, which makes up 28% of the customer base, but consists only of 6% of the overall population of Germany, being wildly overrepresented.

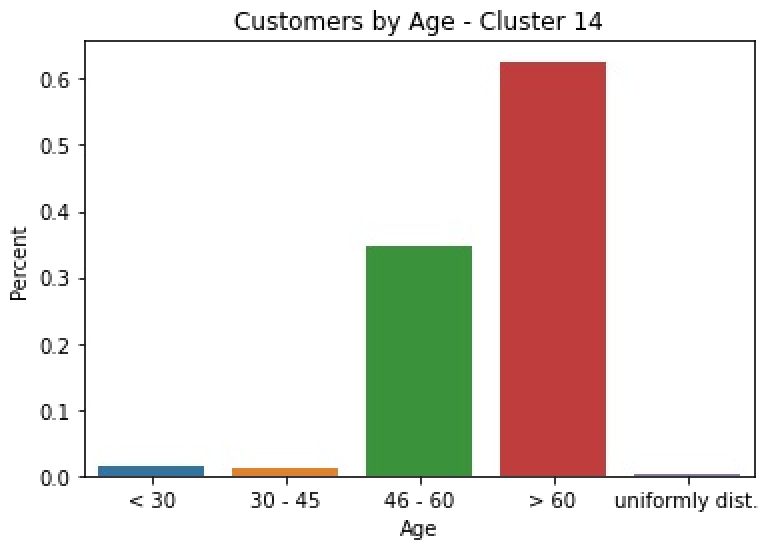


Figure 34: Customers by Age of Customer Segment 14

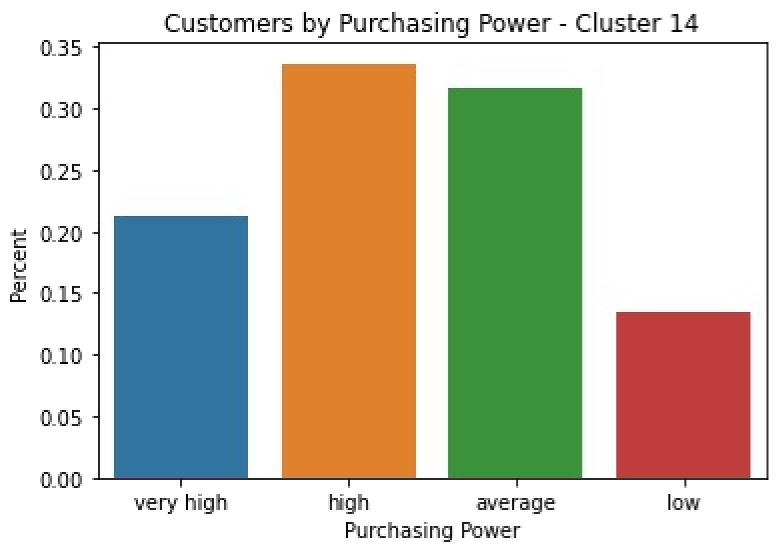


Figure 35: Customers by Purchasing Power of Customer Segment 14

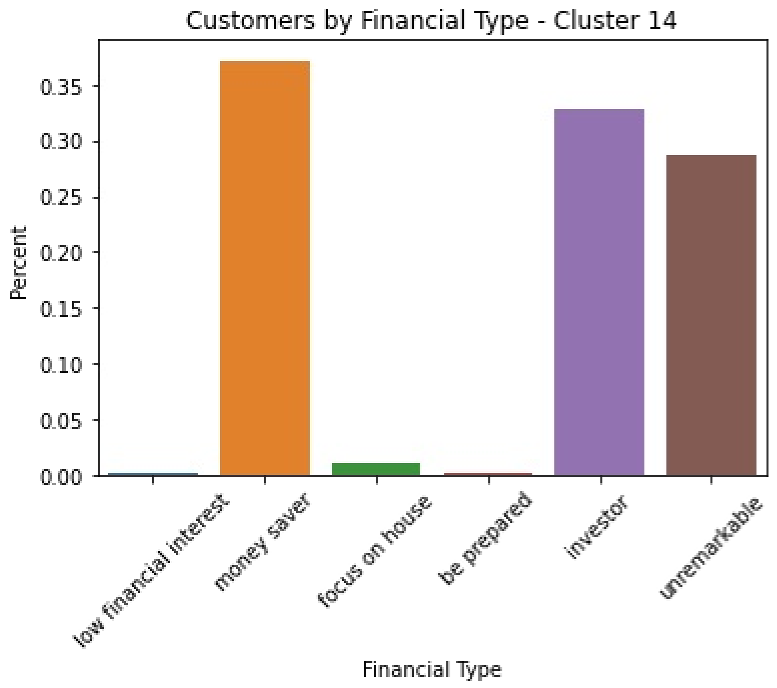


Figure 36: Customers by Financial Type of Customer Segment 14

Based on the three previous figures, we can conclude that this overrepresented cluster consists of older people -- 46 years old or up, and barely anyone under that, with a large concentration of people over 60 years of age. They are also pretty well off, with a majority reporting high or very high purchasing power, and considered of a “Money Saver,” “Investor,” or “Unremarkable” financial type. This tracks with our expectation of people you’d want as customers, and who’d have money to spend -- as younger people are usually starting careers, are have little disposable income to spend, and are less financially savvy.

The next three figures explore Cluster 3, which accounts for only 0.31% of the customer base. This is not the smallest customer segment(that would be Cluster 18 with 0.2%) but it is the most underrepresented when compared with the size of this cluster within the General Population: only of 6% of the overall population of Germany: 6.19%, for an important difference of 5.88%.

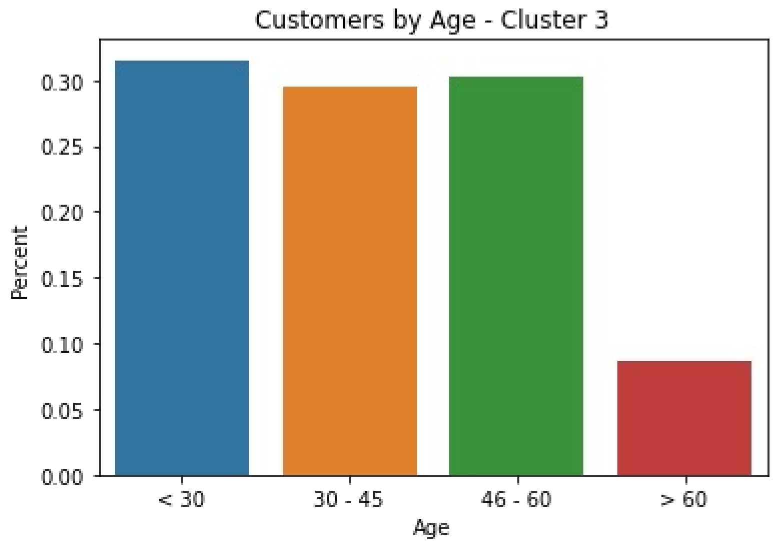


Figure 37: Customers by Age of Customer Segment 3

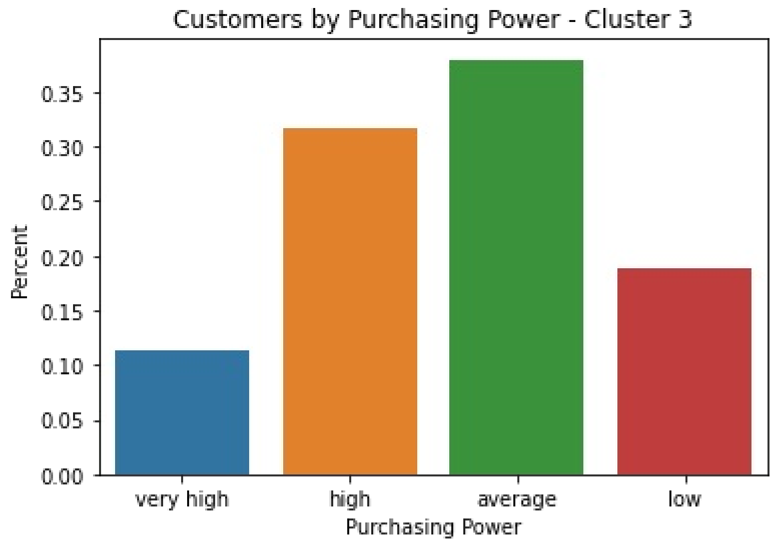


Figure 38: Customers by Purchasing Power of Customer Segment 3

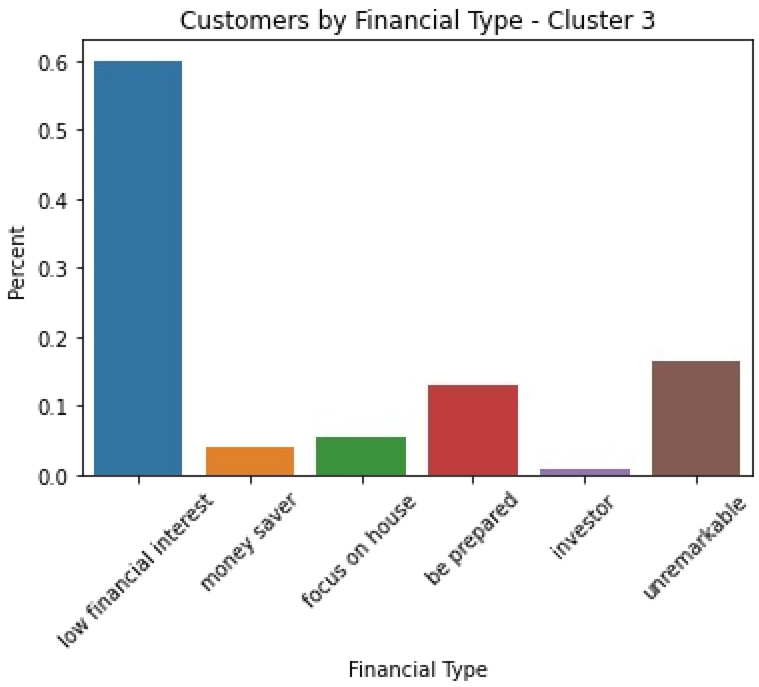


Figure 39: Customers by Financial Type of Customer Segment 3

Based on the three previous figures, we get almost the opposite picture of Cluster 14. The underrepresented cluster consists of people almost uniformly of all ages, but we see the <30 and 30-45 buckets are considerably higher, with those over 60 being a decisive minority. Again, less skewed purchasing power, but most of these individuals are average, with low purchasing power growing and high purchasing power shrinking. Then there's a pretty big tilt towards individuals with low financial interest, who barely exist in the previous cluster.

Again, very different from the overrepresented customer segment, and the reasons are clear: these are younger, less well off individuals who are not as interested in their finances. This makes sense in terms of who would not be part of a traditional customer base, since they probably don't have much disposable income to spend, and it's probably not a good idea to target this segment as most won't be able to become customers.

Supervised Learning Model

As mentioned in the previous sections, the models was were trained train data and evaluated on the test data created through a 80-20% stratified test-train split.

Cross validation techniques were utilized to short list XGBoost as the best model for this scenario in order to prevent overfitting from impacting model choice. Cross validation permits us to estimate the generalization error of the model when trying to predict on data it has never seen, by training the model several times on different folds and evaluating on validation data not used during training.

To combat the imbalanced nature of the dataset, an oversampling technique ADASYN was applied to the data prior to training.

To tune the model's hyperparameter, Grid Search was utilized alongside cross validation to explore different alternatives and arrive at the optimal hyperparameter values forbeste performance without sacrificing overfitting. As mentioned in the Metrics section, ROC AUC score was utilized as the metric of choice throughout to evaluate the model, creating a baseline of 0.5 for a purely random classifier to evaluate against.

After fitting the final model to the training data, and predicting on the test data and comparing with the test labels, the final area under the ROC curve score was 0.74.

Conclusion

Reflection

This report presented the capstone project for Udacity's Data Science Nanodegree program, working with data provided by Arvato Financial Solutions, a Bertelsmann subsidiary.

The project focused on helping a mail-order sales company in Germany interested in identifying segments of the general population to target with their marketing in order to grow. The project utilized datasets with demographics from both the general population of Germany as well as customers of the mail-order company, with the objective to build a model, using both unsupervised and supervised learning techniques, that identified which individuals are most likely to become customers of the mail-order company.

This project involved:

- Exploring and visualizing the demographics data available for the general population and existing customers.
- Cleaning, transforming, and preparing the data for modeling.
- Performing unsupervised learning techniques to cluster the data into customer segments.
- Analyzing customer segments to describe the relationship between the demographics of the company's existing customers and the general population.
- Building a prediction model with supervised learning techniques targeting customers for a mail-out campaign.
- Iterating and refining the model by trying different algorithms and using cross-validation.
- Evaluating and validating the final model's performance.

The first part of the project utilized unsupervised learning techniques, from Principal Component Analysis for dimensionality reduction to K-Means clustering, resulting in 20 customers segments. Two of this customer segments were analyzed to identify the population clusters most overrepresented and underrepresented among the company's customers.

The second part involved building a supervised learning model using an Extreme Gradient Boosting Classifier to predict who the mail-order can successfully target into becoming customers through a campaign. After several rounds of cross validation and tuning, the final model resulted in an area under the ROC curve score of 0.74.

This was a great opportunity to work with real-life, messy data, which needed cleaning and preprocessing in order to be ready for analysis and modeling. In particular, as an imbalanced dataset, it was an opportunity to try out ADASYN, and oversampling technique I had not had the opportunity to use in the past.

All in all, this project allowed me to practice every step in the data science pipeline, from data analysis and visualization, to data cleaning and preparation, to modeling utilizing both supervised and unsupervised learning techniques. In addition, as an actual business problem, writing this report was a great exercise in analyzing real data and results, and providing clear technical context and conclusions.

Improvement

As with most data science projects, the greatest barrier to more improvements is time.

Several hours were invested in cleaning and preparing the data for analysis and modeling, and then running computationally expensive models and tuning techniques.

With more time, I would focus on improving the final model through the following ideas:

- **Focus more on feature selection and feature engineering:** The datasets precedented by the project came in a language I'm not familiar with and contained lots of glaring issues like missing data. Spending even more time analyzing the data available and making sense of its meaning, and using this information to create better data preparation, feature selection, and feature engineering would undoubtedly result in a higher quality model.
- **Explore new ways to deal with imbalanced data:** Trying out different over- and under-sampling techniques, like SMOTE, could be worthwhile and help training on a skewed dataset. I used ADASYN as it seemed a quick, generic solution that would give me good results, but deeper research into this topic could be a good time investment in the future.
- **Experiment with ensemble models:** I used several basic classifiers, from random forests to logistic regression, to try and pick the best performer quickly. Building an ensemble model combining several classifier, or using more complex models, could result in better performance.
- **Try out other hyperparameter tuning techniques:** I used straight up grid search to tune the model's hyperparameters because of time constraints and familiarity with the process. However, better results might be achievable with other tuning techniques like Randomized Search of Bayesian optimization through the Scikit-Optimize library.

The approach undertaken throughout this project was a bit more straight forward than the suggestions above, but also it allowed for a quick runthrough of the complete data science pipeline. Spending more time on each of the outline proposals, however, would surely increase the final ROC AUC score, giving more certainty that the model could be of use to the mail-order company when it comes to gaining more customers through accurate targeting.