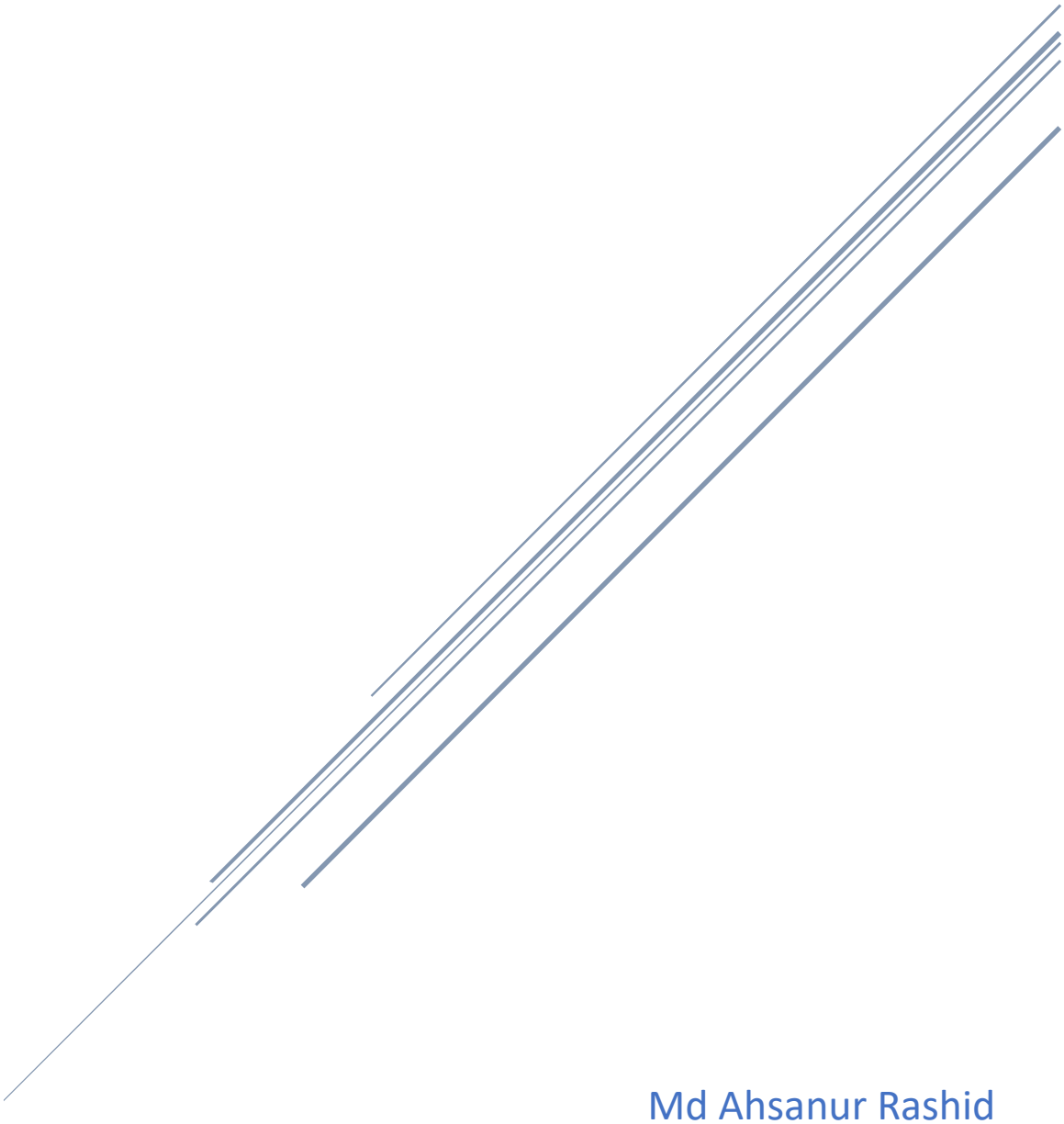
ARVATO CUSTOMER SEGMENTATION AND CAMPAIGN CONVERSION

Machine Learning Nanodegree



Md Ahsanur Rashid
07/29/2020

I. Definition

Project Overview

Direct marketing or direct response marketing was first coined in 1958 [6]. Response channel has evolved over time from reply cards, forms and 800-numbers to websites and emails. Optimization is the key driving force behind every operation today. Advertisement campaigns are no different. There used to be a time when mass advertisement campaigns were all the rage. But with changing times, optimized, targeted advertisement campaigns are more popular. Banks have a lot of information on their clients and it is easy to use that information to learn key characteristics about clients and be able to target persons with a higher chance of winning over with a direct marketing campaign.

This project deals with this optimization of choosing mail order audience with a higher response probability. We have certain information about persons who respond to targeted advertisements and persons who do not. Using this data, we need to come up with a model to predict whether someone is a good candidate to respond to targeted advertisement. We also have certain data about customers and the general population. We use these data to compare the different characteristics of customers vs general population.

Problem Statement

- We need to better understand the customers of Avrato. This can be done by using unsupervised learning and grouping customers into different buckets based on their features.

- We need to create a model to predict whether a person with a given set of features will be a good direct marketing campaign customer or not. This can be done by using supervised learning.

The steps that are needed:

- Clean and process customer and population data
- Use unsupervised learning to cluster customers and general population based on features
- Use supervised learning on a different set of data and create a model for predicting good customers for mail-order campaign
- Test model accuracy

Metrics

For the customer segmentation part, there is no scoring system. But analyzing the cluster centroid data should match with general intuition.

For the customer prediction part, I will use the same evaluation criteria as Kaggle since I do not have access to the expected output for the MailOrder_Test dataset. Kaggle uses AUC for the ROC cure as the evaluation metric for this problem set.

“AUC stands for ‘Area under the ROC Curve’. This measures the entire two-dimensional area underneath the entire ROC curve from (0, 0) to (1, 1). AUC provides an aggregate measure of performance across all possible classification thresholds.” [7] “The ROC curve is plotted with TPR(True Positive Rate) against the FPR (False Positive Rate) where TPR is on y-axis and FPR is on the x-axis. It tells how much a model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.” [8]

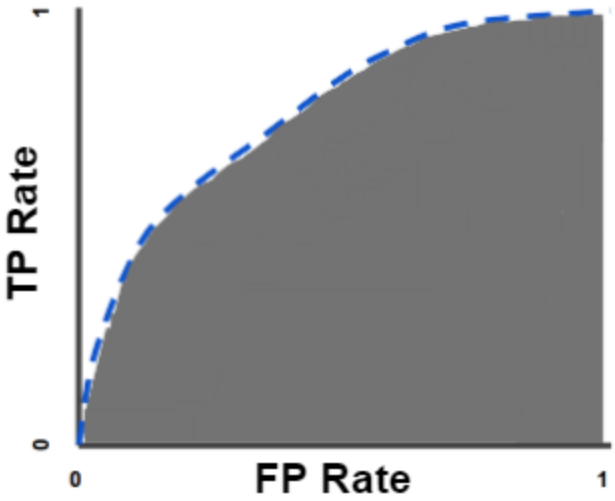


Fig 1: AUC (Area under the ROC Curve) [7]

II. Analysis

Data Exploration

Exploring the data, I looked at the data sets and different columns & the values they hold. I also looked at the distribution of NaN values.

I had four datasets for this part of the project:

a. Azdias dataset

   This is the dataset for the general population in Germany. There are data for 891221 persons, each with 366 attributes.

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALT
0	910215	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	910220	NaN	9.0	NaN	NaN	NaN	NaN	NaN	21.0	
2	910225	NaN	9.0	17.0	NaN	NaN	NaN	NaN	17.0	
3	910226	2.0	1.0	13.0	NaN	NaN	NaN	NaN	13.0	
4	910241	NaN	1.0	20.0	NaN	NaN	NaN	NaN	14.0	
5	910244	3.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
6	910248	NaN	9.0	NaN	NaN	NaN	NaN	NaN	NaN	
7	910261	NaN	1.0	14.0	NaN	NaN	NaN	NaN	14.0	
8	645145	NaN	9.0	16.0	NaN	NaN	NaN	NaN	16.0	
9	645153	NaN	5.0	17.0	NaN	NaN	NaN	NaN	17.0	
10	645165	0.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
11	645169	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
12	612558	NaN	5.0	21.0	NaN	NaN	NaN	NaN	14.0	
13	612561	NaN	8.0	20.0	NaN	NaN	NaN	NaN	20.0	
14	612565	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Table 1: Brief view of Azdias dataset

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FE
count	8.912210e+05	213718.000000	817722.000000	580954.000000	81058.000000	29499.000000	6170.000000	1205.000000	628274.0000
mean	6.372630e+05	1.675376	4.421928	15.291805	11.745392	13.402658	14.476013	15.089627	13.7007
std	2.572735e+05	0.742250	3.638805	3.800536	4.097660	3.243300	2.712427	2.452932	5.0798
min	1.916530e+05	0.000000	1.000000	1.000000	2.000000	2.000000	4.000000	7.000000	0.0000
25%	4.144580e+05	1.000000	1.000000	13.000000	8.000000	11.000000	13.000000	14.000000	11.0000
50%	6.372630e+05	2.000000	3.000000	16.000000	12.000000	14.000000	15.000000	15.000000	14.0000
75%	8.600680e+05	2.000000	9.000000	18.000000	15.000000	16.000000	17.000000	17.000000	17.0000
max	1.082873e+06	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.0000

Table 2: Azdias dataset descriptive statistics

b. Customers dataset

This is the dataset for Avrato customers. There are data for 191652 customers, each with 369 attributes.

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALT
0	9626	2.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
1	9628	NaN	9.0	11.0	NaN	NaN	NaN	NaN	NaN	
2	143872	NaN	1.0	6.0	NaN	NaN	NaN	NaN	0.0	
3	143873	1.0	1.0	8.0	NaN	NaN	NaN	NaN	8.0	
4	143874	NaN	1.0	20.0	NaN	NaN	NaN	NaN	14.0	
5	143888	1.0	1.0	11.0	NaN	NaN	NaN	NaN	10.0	
6	143904	2.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
7	143910	1.0	1.0	10.0	NaN	NaN	NaN	NaN	9.0	
8	102160	2.0	3.0	5.0	NaN	NaN	NaN	NaN	4.0	
9	102173	1.0	1.0	20.0	NaN	NaN	NaN	NaN	13.0	
10	102184	NaN	7.0	14.0	NaN	NaN	NaN	NaN	14.0	
11	102185	1.0	1.0	10.0	NaN	NaN	NaN	NaN	10.0	
12	102227	NaN	1.0	21.0	NaN	NaN	NaN	NaN	14.0	
13	102230	NaN	1.0	15.0	8.0	NaN	NaN	NaN	14.0	
14	102239	2.0	1.0	6.0	NaN	NaN	NaN	NaN	6.0	

Table 3: Brief view of Customers dataset

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FE
count	191652.000000	99545.000000	145056.000000	122905.000000	11766.000000	5100.000000	1275.000000	236.000000	139810.0000
mean	95826.500000	1.588267	1.747525	13.397966	12.337243	13.672353	14.647059	15.377119	10.3315
std	55325.311233	0.713589	1.966334	4.365868	4.006050	3.243335	2.753787	2.307653	4.1348
min	1.000000	0.000000	1.000000	2.000000	2.000000	2.000000	5.000000	8.000000	0.0000
25%	47913.750000	1.000000	1.000000	10.000000	9.000000	11.000000	13.000000	14.000000	9.0000
50%	95826.500000	2.000000	1.000000	13.000000	13.000000	14.000000	15.000000	16.000000	10.0000
75%	143739.250000	2.000000	1.000000	17.000000	16.000000	16.000000	17.000000	17.000000	13.0000
max	191652.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.0000

Table 4: Azdias dataset descriptive statistics

c. MailOut_Train dataset

This is the dataset for persons that responded positively or negatively to targeted advertisement. There are data for 42,962 persons, each with 367 attributes including their response.

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	AI
count	42962.000000	42962.000000	35993.000000	35993.000000	1988.000000	756.000000	174.000000	41.000000	34807.000000	
mean	42803.120129	0.542922	1.525241	10.285556	12.606137	13.783069	14.655172	14.195122	9.855058	
std	24778.339984	1.412924	1.741500	6.082610	3.924976	3.065817	2.615329	3.034959	4.373539	
min	1.000000	-1.000000	1.000000	0.000000	2.000000	5.000000	6.000000	6.000000	0.000000	
25%	21284.250000	-1.000000	1.000000	8.000000	9.000000	12.000000	13.000000	13.000000	8.000000	
50%	42710.000000	1.000000	1.000000	10.000000	13.000000	14.000000	15.000000	15.000000	10.000000	
75%	64340.500000	2.000000	1.000000	15.000000	16.000000	16.000000	17.000000	17.000000	13.000000	
max	85795.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.000000	

Table 5: Brief view of MailOut_Train dataset

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALTE
0	1763	2	1.0	8.0	NaN	NaN	NaN	NaN	8.0	
1	1771	1	4.0	13.0	NaN	NaN	NaN	NaN	13.0	
2	1776	1	1.0	9.0	NaN	NaN	NaN	NaN	7.0	
3	1460	2	1.0	6.0	NaN	NaN	NaN	NaN	6.0	
4	1783	2	1.0	9.0	NaN	NaN	NaN	NaN	9.0	
5	1789	3	1.0	12.0	NaN	NaN	NaN	NaN	12.0	
6	1795	1	1.0	8.0	NaN	NaN	NaN	NaN	8.0	
7	1493	2	1.0	13.0	NaN	NaN	NaN	NaN	13.0	
8	1801	-1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9	1834	-1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
10	1838	-1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
11	2512	2	1.0	8.0	NaN	NaN	NaN	NaN	8.0	
12	2513	1	1.0	15.0	NaN	NaN	NaN	NaN	8.0	
13	2515	1	1.0	20.0	NaN	NaN	NaN	NaN	12.0	
14	2198	-1	1.0	21.0	13.0	NaN	NaN	NaN	16.0	

Table 6: MailOut_Train dataset statistics

d. MailOut_Test dataset

This is the test dataset for prediction models created using the MailOut_Train dataset. There are data for 42,833 persons, each with 366 attributes (doesn't include their response).

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALTE
0	1754	2	1.0	7.0	NaN	NaN	NaN	NaN	6.0	
1	1770	-1	1.0	0.0	NaN	NaN	NaN	NaN	0.0	
2	1465	2	9.0	16.0	NaN	NaN	NaN	NaN	11.0	
3	1470	-1	7.0	0.0	NaN	NaN	NaN	NaN	0.0	
4	1478	1	1.0	21.0	NaN	NaN	NaN	NaN	13.0	
5	1782	2	1.0	7.0	NaN	NaN	NaN	NaN	7.0	
6	1485	2	1.0	10.0	NaN	NaN	NaN	NaN	9.0	
7	1519	-1	1.0	20.0	NaN	NaN	NaN	NaN	15.0	
8	1835	1	1.0	19.0	NaN	NaN	NaN	NaN	13.0	
9	1522	1	1.0	0.0	NaN	NaN	NaN	NaN	9.0	
10	1539	0	1.0	0.0	NaN	NaN	NaN	NaN	12.0	
11	1853	1	1.0	6.0	NaN	NaN	NaN	NaN	6.0	
12	1856	-1	2.0	20.0	NaN	NaN	NaN	NaN	19.0	
13	2502	2	1.0	11.0	NaN	NaN	NaN	NaN	11.0	
14	2182	2	1.0	7.0	NaN	NaN	NaN	NaN	7.0	

Table 7: Brief view of MailOut_Test dataset

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	AI
count	42833.000000	42833.000000	35944.000000	35944.000000	2013.000000	762.000000	201.000000	39.000000	34715.000000	
mean	42993.165620	0.537436	1.518890	10.239511	12.534029	13.942257	14.442786	14.410256	9.822584	
std	24755.599728	1.414777	1.737441	6.109680	3.996079	3.142155	2.787106	2.279404	4.410937	
min	2.000000	-1.000000	1.000000	0.000000	2.000000	4.000000	6.000000	9.000000	0.000000	
25%	21650.000000	-1.000000	1.000000	8.000000	9.000000	12.000000	13.000000	13.000000	8.000000	
50%	43054.000000	1.000000	1.000000	10.000000	13.000000	14.000000	15.000000	14.000000	10.000000	
75%	64352.000000	2.000000	1.000000	15.000000	16.000000	17.000000	17.000000	16.000000	13.000000	
max	85794.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.000000	

Table 8: MailOut_Test dataset statistics

There were two additional excel files to help understand the data:

- DIAS Information Levels – Attributes 2017.xlsx

    - Description and clustering of related features
- DIAS Attributes – Values 2017.xlsx
    - Possible values of each feature and their meaning

There are many imperfections in the given datasets. There is quite a bit of NaN entries (both row and column wise) that need to be processed. There are a few columns with categorical data that need to be hot encoded. Some columns have mixed data types. Some of the data are skewed and need to be dropped. Outliers are present as well.

Exploratory Visualization

ANDREDE_KZ or gender column caught my attention since it seemed like there was either male or unknown in the values. No females. In the dataset:

- -1, 0 : unknown
- 1 : male
- 2 : female

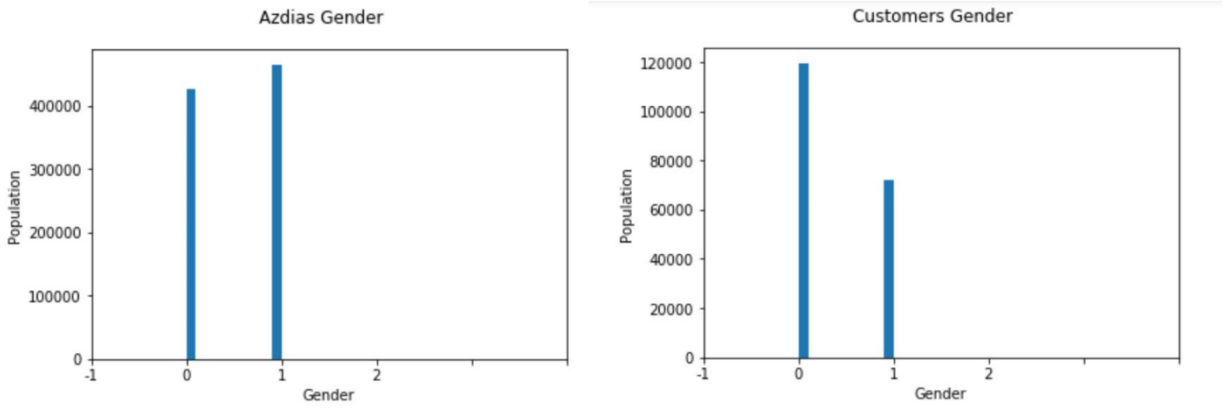


Fig 2: Value distribution for ANDREDE_KZ (gender) in Azdias dataset and Customers dataset

Another interesting one was LNR which had a different value for every row. I ended up dropping both LNR and ANDREDE_KZ columns.

```
0 ColumnName:  LNR UniqueLength 191652
Unique Values:  [  9626   9628 143872 ... 148813 148852 148883]

18 ColumnName:  CAMEO_DEUG_2015 UniqueLength 11
Unique Values:  ['1' nan '5' '4' '7' '3' '9' '2' '6' '8' 'X']
19 ColumnName:  CAMEO_INTL_2015 UniqueLength 23
Unique Values:  ['13' nan '34' '24' '41' '23' '15' '55' '14' '22' '43' '51
' '33' '25' '44' '54' '32' '12' '35' '31' '45' '52' 'XX']

367 ColumnName:  ANREDE_KZ UniqueLength 2
Unique Values:  [1 2]
```

Fig 3: Different possible and total values for the columns LNR, CAMEO_DEUG_2015, CAMEO_INTL_2015 and ANREDE_KZ

Another characteristic worth mentioning is the number of missing data denoted by NaNs. For some columns this is as high as ~100%.

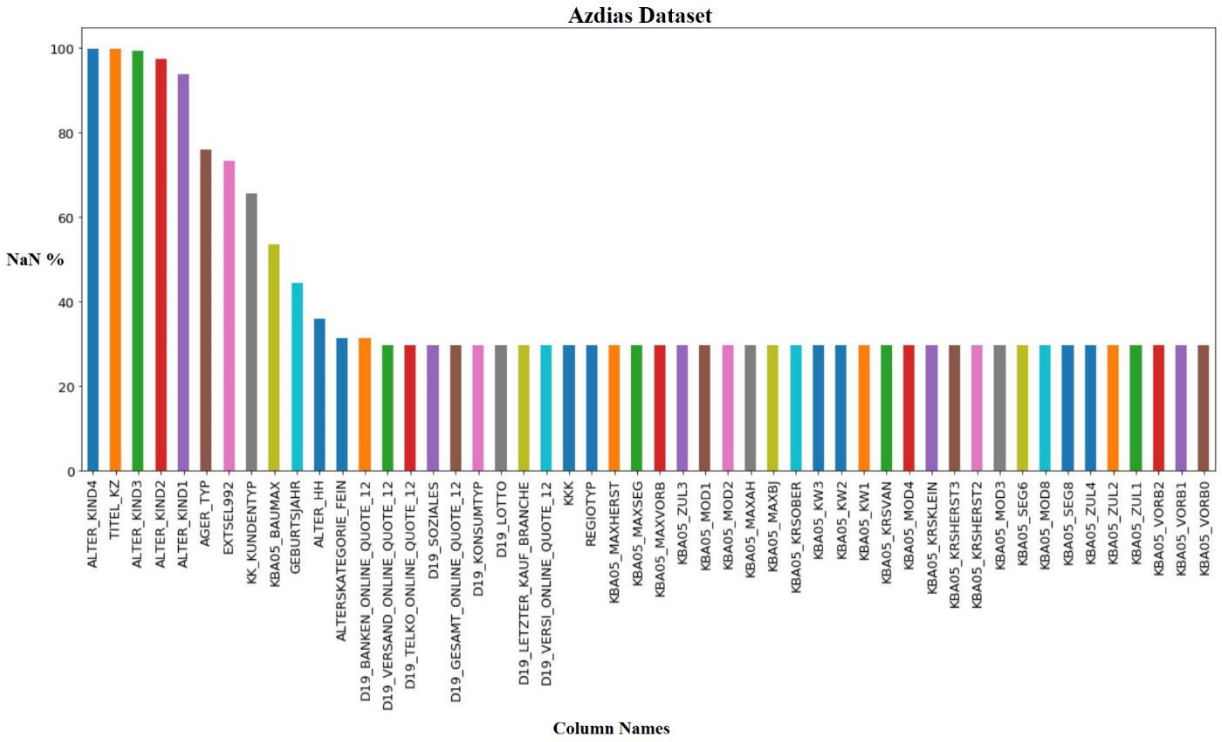


Fig 4: Azdias dataset NaN distribution

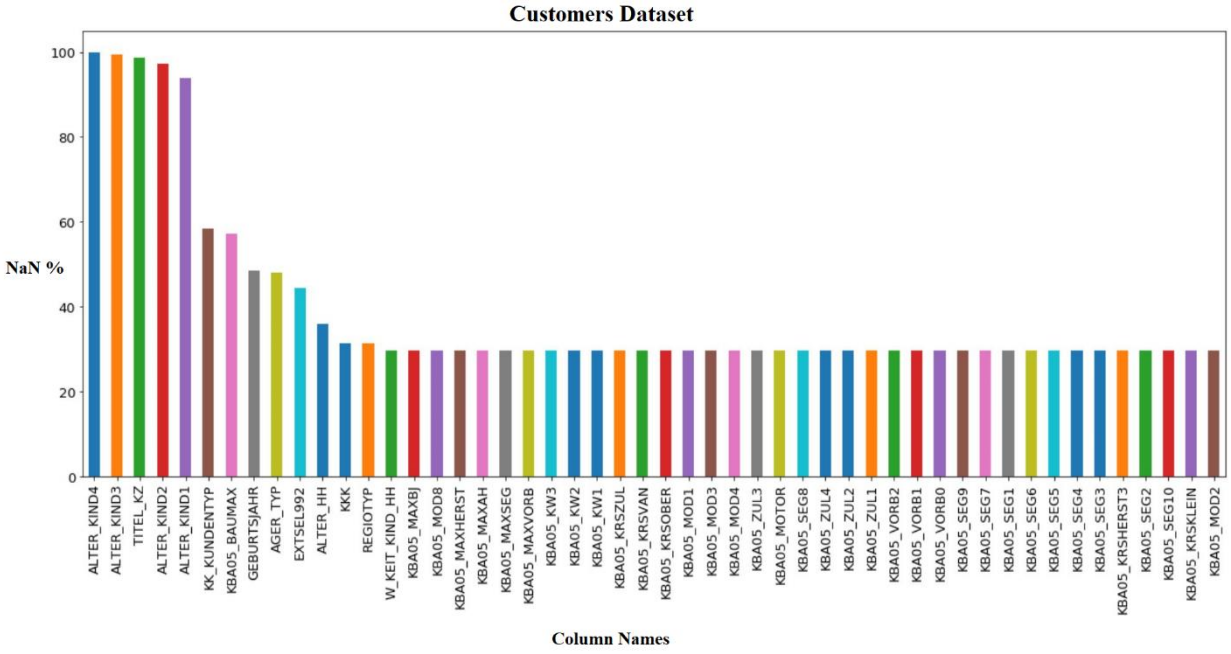


Fig 5: Customers dataset NaN distribution

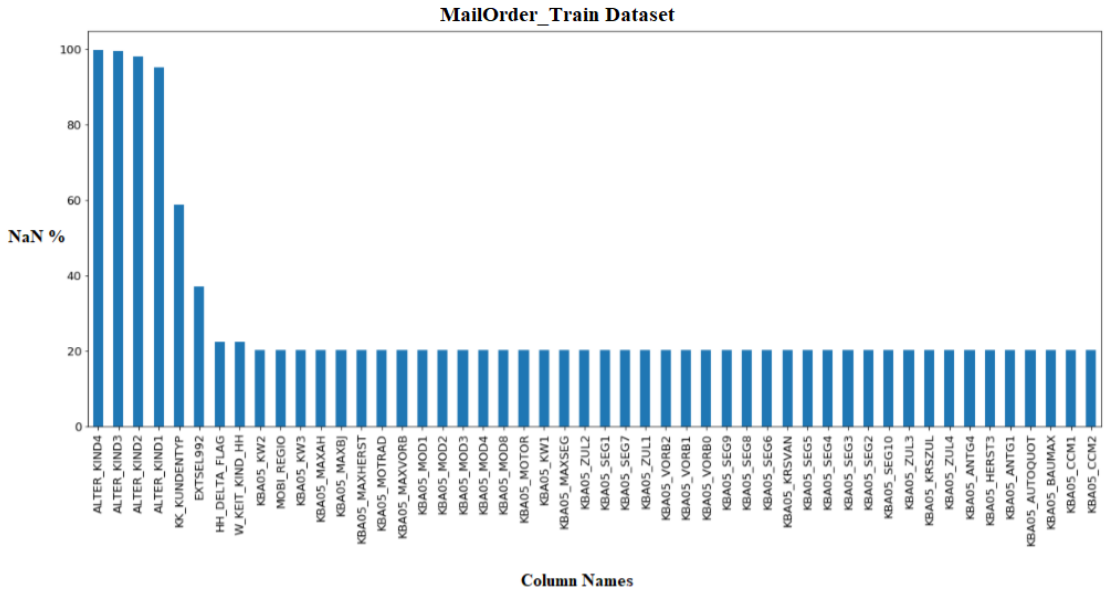


Fig 6: MailOrder_Train dataset NaN distribution

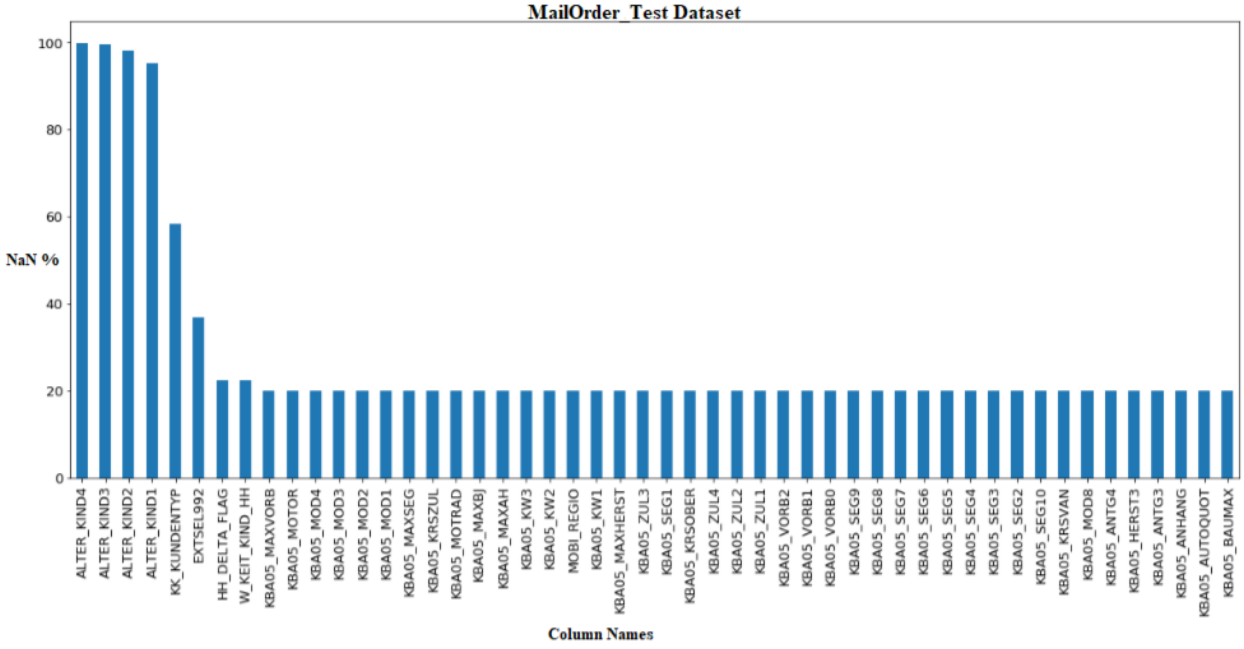


Fig 7: MailOrder_Test Dataset NaN distribution

For the MailOrder_Test dataset, I noticed very few positive (1) responses and an overwhelming number of negative (0) responses. This will make it harder to create a model and validate.

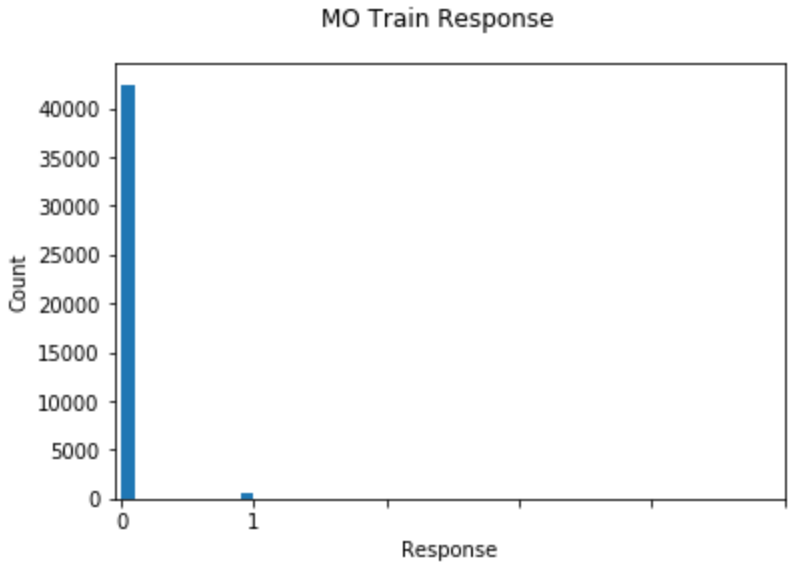


Fig 8: Response distribution for MailOrder_Train dataset

Algorithms and Techniques

- **Principal Component Analysis**

  Principal Component Analysis or PCA is a dimensionality reduction method that is used to reduce the dimensionality of large data sets, by transforming a large set of features into a smaller one that still contains most of the information in the large set. [1] PCA is done either by singular value decomposition of a design matrix or by calculating the correlation or covariance matrix and performing eigenvalue decomposition on that. [2]

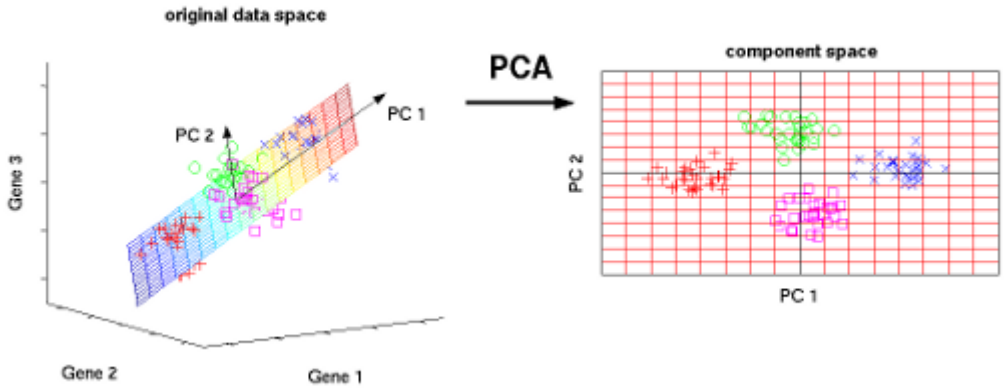


Fig 9: Principal Component Analysis [4]

- **K-Means Clustering**

  K-means clustering is an iterative algorithm that aims to partition the dataset into k pre-defined, distinct, non-overlapping clusters where each data point belongs to only one group. [3] It also tries to make the intra-cluster data points as similar as possible while keeping the clusters as different (far) as possible.

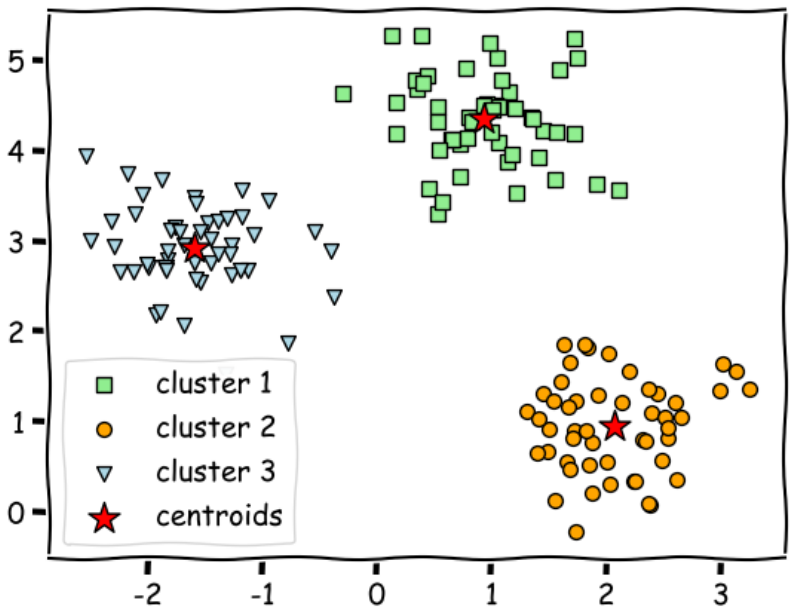


Fig 10: K-Means Clustering [5]

- **Linear Regression**

  Linear Regression is a kind of regression analysis in which there is a relation between one or more independent variable and one dependent variable. E.g. y = ax + b. The goal is to minimize the cost function ($\frac{1}{n}\sum_{i=1}^{n} square(\text{predictedYi} - \text{Yi})$) by selecting and changing the values of a and b. [9]

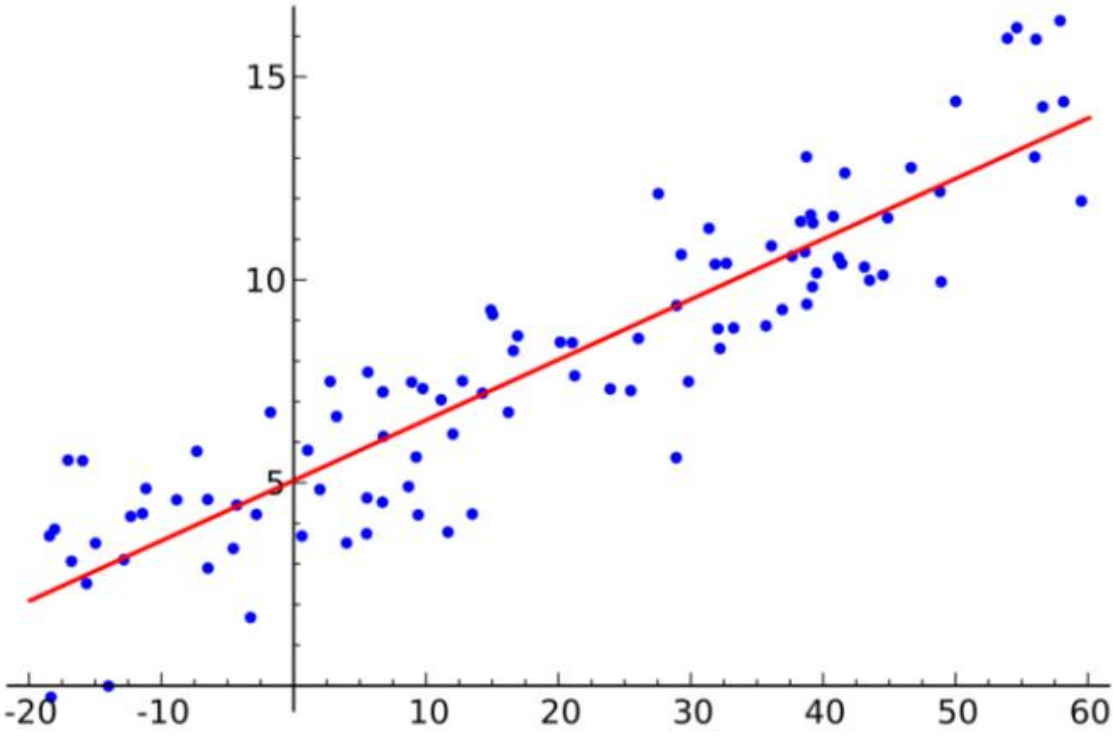


Fig 11: Linear Regression [9]

- **XGBoost**

  XGBoost is a tree-based learning algorithm. This is one of the best algorithms for small to medium structured data. [10]

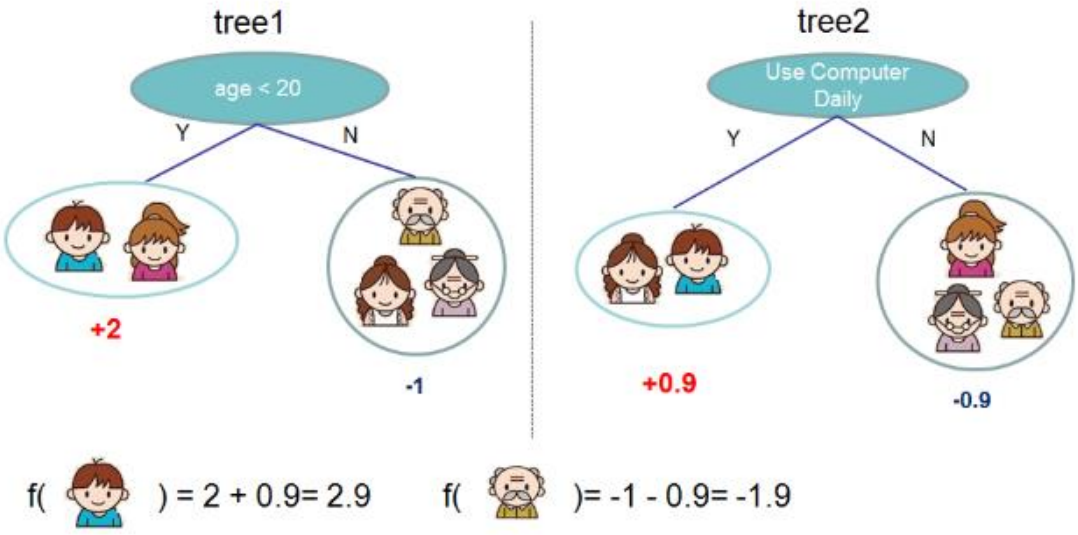


Fig 12: XGBoost/GBM Learner (Tree) Ensemble [12]

In Gradient Boosted Models, an ensemble of weak learner (trees) are built with greater weight given to misclassified labels. These learners are then combined to form a stronger learner (tree). These trees are created sequentially. The weights in a tree model are derived from Gradient Descent to optimize the cost function. GBM combines a new function with existing function at each step. Essentially, GBM takes a weighted sum of multiple models. [12]

XGBoost is a major improvement on Gradient Boosted Models (GBM). One of the main differences is that the learner or tree building is parallelized in XGBoost. Features like regularization, cache-aware access, sparsity-awareness, out-of-core computation, column block for parallel learning etc. make XGBoost scalable in distributed and memory-limited settings. [12]

- **Techniques**
    - Since this is a rather large project and I had to use a lot of code, I created functions and stored them in a separate cleanupMethods.py file. This way, it was easy to work on either part of the project without going through too many initialization steps.

- Given the customer and population distribution post PCA and K-Means, I chose a couple of clusters to investigate. For these clusters, I got the cluster centers. Then, I inverse transformed the principal component of the cluster centers to the original components. Then I performed another inverse transformation using the standard scaler. These mapped the cluster center in the k-means clustering algorithm output to the original column values in the cleaned dataset. This helped me get a better understanding of what made up each cluster.

Benchmark

As a reference point, there is a leaderboard on Kaggle for a competition on the same dataset for customer prediction. The current #1 has a score of 0.81063 and #150 has a score of 0.74393. I use these scores as a benchmark.

III. Methodology

Data Preprocessing

Data cleanup is the most important & time-consuming part of any ML work and this was no different. I had to go through the data, explore it and figure out different techniques for cleanup. Clean up was done on all data set. I did the following, in order, to clean the data before doing PCA analysis and KMeans clustering:

- Marked some data fields as NaN

Based on the provided attributes and probable values documented, many fields had -1, 0, 9 etc. for unknown values and were marked down as NaN for further processing later.

- Cleaned up mixed type (float and string) columns
    - CAMEO_INTL_2015
    - CAMEO_DEUG_2015

  The values ‘X’ and ‘XX’ were replaced by NaN.

- Removed differences between the Customers and Azdias data set columns
    - CUSTOMER_GROUP
    - ONLINE_PURCHASE
    - PRODUCT_GROUP

  Removed the extra columns from the Customers data set.

- Dropped columns where 33.33% of the data were NaN
    - ALTER_KIND4
    - ALTER_KIND3
    - TITEL_KZ
    - ALTER_KIND2
    - ALTER_KIND1
    - KK_KUNDENTYP
    - KBA05_BAUMAX
    - GEBURTSJAHR
    - AGER_TYP
    - EXTSEL992
    - ALTER_HH

  This was decided by looking at the percentage of missing data for each column. Removing the top 11 columns with the most missing data. This was selected based on how many columns need to be dropped and what percentage of missing data seemed

feasible. This also happened to overall drop the same columns for both Azdias and Customers data set. Dropped mostly similar columns for MailOrder_Train and MailOrder_Test datasets.

- Removed columns with mostly unique values
    - LNR

  Looked at the different types of values for each column and noticed LNR column had unique values for each row. Deleted the column from all data sets for training and validation.

- Dropped columns where data is clearly skewed
    - ANREDE_KZ

  While exploring the data set it was noticed that the values for the column were either male or unknown. Dropped from all data sets.

- Dropped undocumented columns
    - AKT_DAT_KL
    - ANZ_STATISTISCHE_HAUSHALTE
    - ARBEIT
    - CJT_KATALOGNUTZER
    - CJT_TYP_1
    - CJT_TYP_2
    - CJT_TYP_3
    - CJT_TYP_4
    - CJT_TYP_5
    - CJT_TYP_6
    - D19_KONSUMTYP_MAX
    - D19_LETZTER_KAUF_BRANCHE
    - D19_SOZIALES
    - D19_TELKO_ONLINE_QUOTE_12
    - D19_VERSI_DATUM

- D19_VERSI_OFFLINE_DATUM
- D19_VERSI_ONLINE_DATUM
- D19_VERSI_ONLINE_QUOTE_12
- DSL_FLAG
- EINGEFUEGT_AM
- EINGEZOGENAM_HH_JAHR
- EXTSEL1992
- FIRMENDICHTE
- GEMEINDETYP
- HH_DELTA_FLAG
- KBA13_ANTG1
- KBA13_ANTG2
- KBA13_ANTG3
- KBA13_ANTG4
- KBA13_BAUMAX
- KBA13_GBZ
- KBA13_HHZ
- KBA13_KMH_210
- KK_KUNDENTYP
- KOMBIALTER
- KONSUMZELLE
- MOBI_RASTER
- RT_KEIN_ANREIZ
- RT_SCHNAEPPCHEN
- RT_UEBERGROESSE
- SOHO_KZ
- STRUKTURTYP
- UMFELD_ALT
- UMFELD_JUNG
- UNGLEICHENN_FLAG
- VERDICHTUNGSRAUM

    - VHA
    - VHN
    - VK_DHT4A
    - VK_DISTANZ
    - VK_ZG11

  Some columns from the list above were dropped already as part of other dropping criteria. Some undocumented columns were not removed, since they were easy to understand and seemed important to keep:

    - ANZ_KINDER

- Dropped columns with too many values
    - CAMEO_DEU_2015
    - D19_LETZTER_KAUF_BRANCHE

  These has 44 and 36 types of values respectively.

- Dropped columns deemed unnecessary
    - MIN_GEBAEUDEJAHR

  Since this represents the year the building was first mentioned in the database, it seemed unnecessary data to analyze.

- Dropped additional columns for Grob vs Fein scenarios.

  There were 4 pairs of columns that had remarkably similar data. One was the FEIN or Fine column and the other one was the GROB or rough column. The fine column had more possible values or buckets and more finely sorted the data. Whereas the rough column had bigger buckets or less number of probable values. Since the fine columns had quite a lot of probable values, decided to drop FEIN columns and keep GROB columns.

    - ALTERSKATEGORIE_FEIN
    - LP_FAMILIE_FEIN

- LP_LEBENSPHASE_FEIN
- LP_STATUS_FEIN

- Dropped all rows with 30% or more NaN values.

  51,281 rows were dropped for Customers dataset, 105,800 rows were dropped for Azdias dataset and 7,955 rows were dropped for MailOrder_Train dataset. MailOrder_Test dataset was reserved for testing and no rows were dropped.

- Binary encoded OST_WEST_KZ and VERS_TYP columns

Column Name	Old Value	New Value
OST_WEST_KZ	W	1
	O	0
VERS_TYP	1	1
	2	0

Table 9: Value mapping for binary encoding of OST_WEST_KZ and VERS_TYP

- Replaced NaN values with median or most frequently used values
  - For binary columns, used most frequently used value to replace NaNs
  - For all other columns used median value to replace NaNs

- Split some columns into multiple columns
  - CAMEO_INTL_2015
  - PLZ8_BAUMAX
  - PRAEGENDE_JUGENDJAHRE
  - WOHNLAGE

  Except for the last two, all of them were dropped and new ones created to replace them.

Old Column Name	Old Value	Meaning	New Column Name	New Value	Meaning
CAMEO_INTL_2015	11	Wealthy Households-Pre-Family Couples & Singles	CI2_Family Type	1	Pre Family Couples & Singles
	12	Wealthy Households-Young Couples With Children		2	Young Couples with Children
	13	Wealthy Households-Families With School Age Children		3	Families with school age children
	14	Wealthy Households-Older Families & Mature Couples		4	Older families & Mature couples
	15	Wealthy Households-Elders In Retirement		5	Elders in retirement
	21	Prosperous Households-Pre-Family Couples & Singles	CI2_Wealth Type	1	Wealthy Households
	22	Prosperous Households-Young		2	Prosperous Households

		Couples With Children			
	23	Prosperous Households-Families With School Age Children		3	Comfortable Households
	24	Prosperous Households-Older Families & Mature Couples		4	Less Affluent Households
	25	Prosperous Households-Elders In Retirement		5	Poorer Households
	31	Comfortable Households-Pre-Family Couples & Singles			
	32	Comfortable Households-Young Couples With Children			
	33	Comfortable Households-Families With School Age Children			
	34	Comfortable Households-Older Families & Mature Couples			

	35	Comfortable Households-Elders In Retirement	
	41	Less Affluent Households-Pre-Family Couples & Singles	
	42	Less Affluent Households-Young Couples With Children	
	43	Less Affluent Households-Families With School Age Children	
	44	Less Affluent Households-Older Families & Mature Couples	
	45	Less Affluent Households-Elders In Retirement	
	51	Poorer Households-Pre-Family Couples & Singles	
	52	Poorer Households-Young Couples With Children	
	53	Poorer Households-Families With School Age Children	

	54	Poorer Households-Older Families & Mature Couples			
	55	Poorer Households-Elders In Retirement			
PLZ8_BAUMAX	1	mainly 1-2 family homes	PB_Family	0	Not mainly family home
	2	mainly 3-5 family homes		1	Mainly family home
	3	mainly 6-10 family homes	PB_Business	0	Not maintly business building
	4	mainly >10 family homes		1	Mainly business building
	5	mainly business building			
PRAEGENDE_JUGENDJAHRE	1	40’s - war years (Mainstream, O+W)	PJ_Movement	0	Mainstream
	2	40’s - reconstruction years (Avantgarde, O+W)		1	Avantgarde
	3	50’s - economic miracle (Mainstream, O+W)	PJ_Generation	1	40’s
	4	50’s - milk bar / Individualisation (Avantgarde, O+W)		2	50’s
	5	60’s - economic miracle (Mainstream, O+W)		3	60’s

	6	60’s - generation 68 / student protestors (Avantgarde, W)		4	70’s
	7	60’s - opponents to the building of the Wall (Avantgarde, O)		5	80’s
	8	70’s - family orientation (Mainstream, O+W)		6	90’s
	9	70’s - peace movement (Avantgarde, O+W)			
	10	80’s - Generation Golf (Mainstream, W)			
	11	80’s - ecological awareness (Avantgarde, W)			
	12	80’s - FDJ / communist party youth organisation (Mainstream, O)			
	13	80’s - Swords into ploughshares (Avantgarde, O)			
	14	90’s - digital media kids (Mainstream, O+W)			

	15	90’s - ecological awareness (Avantgarde, O+W)			
WOHNLAGE	1	very good neighbourhood	WL_Rural	0	Not rural
	2	good neighbourhood		1	Rural
	3	average neighbourhood			
	4	poor neighbourhood			
	5	very poor neighbourhood			
	7	rural neighbourhood			
	8	new building in rural neighbourhood			

Table 10: Value mapping of newly split columns

- Converted to integer values

  Converted all the values to integer values for all the data sets.

- Removed dataset outliers

  For all rows, for each columns with non-binary value, values outside a +/- 6 difference with standard deviation were dropped.

  17,353 rows were dropped from Customers data set, 82,904 rows were dropped from Azdias data set and 94 rows were dropped from MailOrder_Train dataset.
  MailOrder_Test dataset was reserved for testing and no rows were dropped.

- Removed columns with low variance

Removed some columns with low variance in Azdias and MailOrder_Train datasets and the corresponding columns were also removed from Customers and MailOrder_Test datasets.

- Scaled dataset values

  Performed standard scaler on all data sets.

Implementation

- **Principal Component Analysis**

  I performed PCA on the cleaned Azdias data set and calculated the cumulative variance for the principal components such that we achieve a certain variance % (in this case 95%). The same principal components were used to transform the Customers data set.

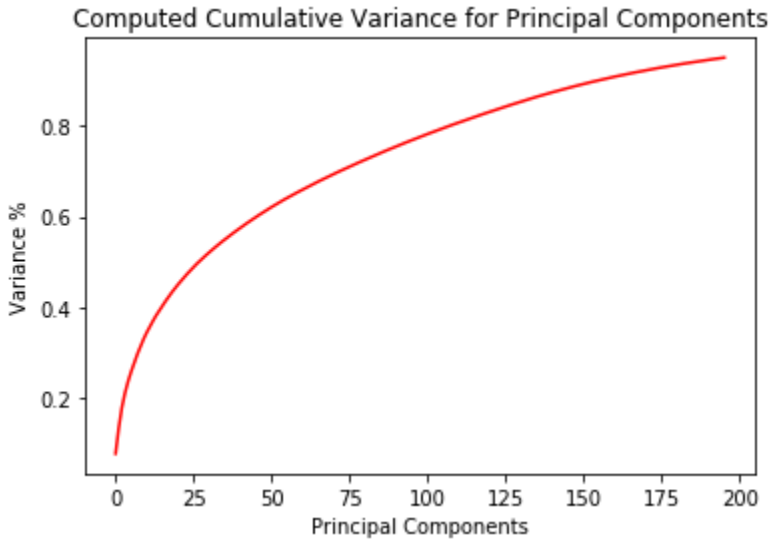


Fig 13: Computed Cumulative Variance for Principal Components

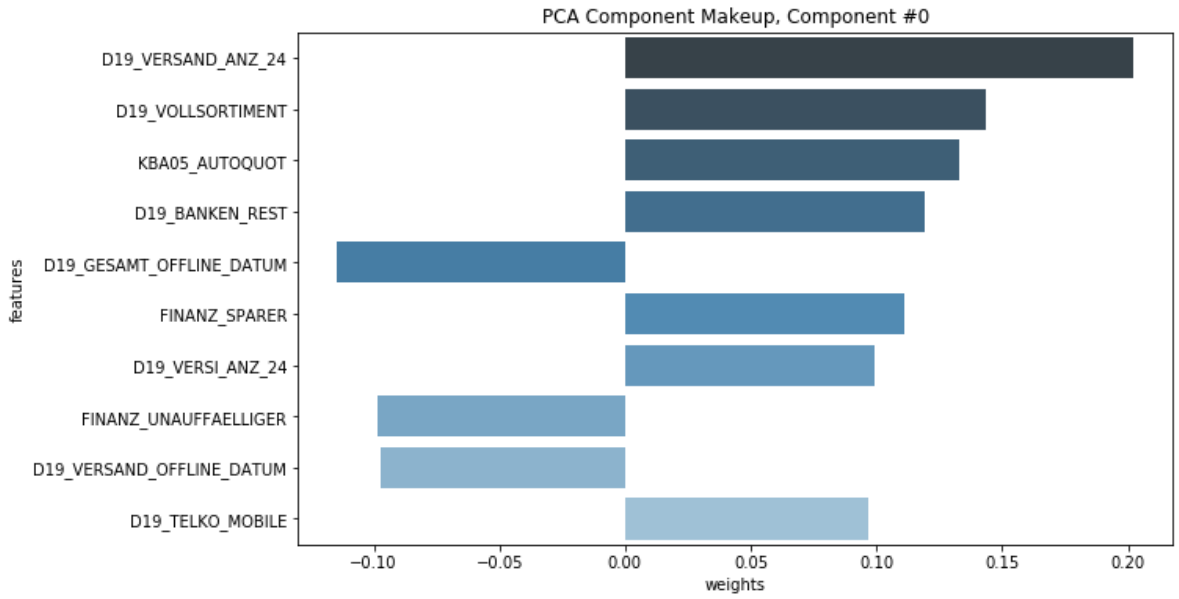


Fig 14: PCA Component Makeup, Component #0

- **K-Means Clustering**

  I did k-means clustering for the Azdias data set using different cluster sizes from 1 to 20. Plotting the cluster size against the model score, I got the elbow chart which helped me choose the cluster size of 10.

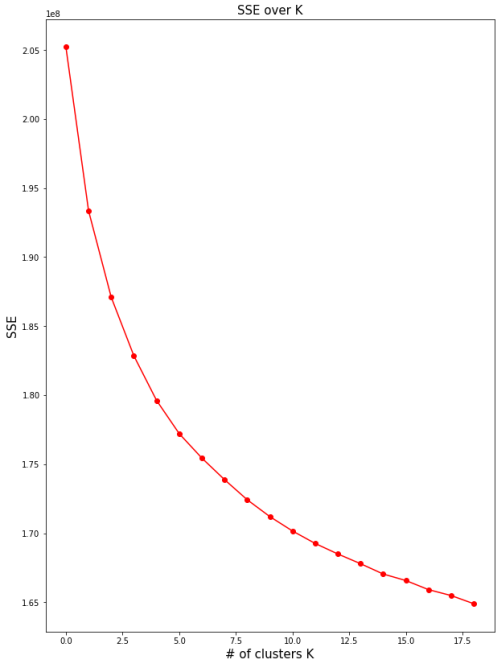


Fig 15: Elbow Chart

	Azdias	Customers	% of processed Azdias	% of processed Customers
0	71778	1819	10.031095	1.439253
1	90890	15403	12.702028	12.187364
2	47323	2806	6.613468	2.220200
3	66218	5344	9.254076	4.228350
4	64821	27365	9.058842	21.652095
5	28607	3336	3.997876	2.639554
6	89864	1127	12.558643	0.891720
7	53235	2453	7.439680	1.940895
8	103936	31124	14.525229	24.626340
9	98883	35608	13.819064	28.174230

Fig 16: Final cluster list

- **Linear Regression**

  I used the linear algorithm with the following major hyperparameters:

  - predictor_type: 'binary_classifier' (since we have only two labels/values)

- binary_classifier_model_selection_criteria: 'accuracy' (since we are using AUC for scoring)
- positive_example_weight_mult = 'balanced' (so that errors in classifying both positive and negative examples have the same weight, since we have imbalanced data.)
- l1: '0.1' (L1 regularization parameter, used to avoid overfitting)
- wd: '0.1' (L2 regularization parameter, used to avoid overfitting)
- use_bias: '(True, False)' (Should a bias term be included? Tuned using both values.)
- mini_batch_size: '(300, 1000)' (Number of observations per mini-batch for the data iterator. Tuned using values in this range.)
- learning_rate: '(0.1, 0.5)' (Step size used by the parameter. Tuned using values in this range.)

- **XGBoost**

  I used the XGBoost algorithm with the following major hyperparameters:
  - alpha: (0, 100) (L1 regularization parameter, used to avoid overfitting. Tuned using the values in this range.)
  - early_stopping_rounds: '10' (Validaton error needs to decrease at least every x rounds to continue training.)
  - eta: (0.1, 0.5) (Step size shrinkage used to prevent overfitting. Tuned using the values in this range.)
  - gamma: (0, 50) (Minimum loss reduction required to make a further partition on a leaf node. The larger the value, the more conservative it is. Tuned using the values in this range.)
  - lambda: (0, 100) (L2 regularization parameter. Increasing this makes the model more conservative. Tuned using the values in this range.)
  - max_depth: (2, 10) (Maximum depth of the tree. Increasing value makes the tree more complex and prone to overfitting. Tuned using the values in this range.)

- min_child_weight: (2, 100) (Minimum sum of weight need in a child. The higher the value, the more conservative is the model. Tuned using the values in this range.)
- num_round: (100, 5000) (Number of rounds to run the training. Tuned using the values in this range.)
- objective: 'binary:logistic' (Since there are only two labels/values)
- subsample: (0.3, 1.0) (Subsample ratio of the training instance, what percentage of data is randomly collected. Tuned using the values in this range.)

Refinement

I initially started by using Linear Regression algorithm. While the score (0.46066) was low, as expected, there seemed to be a row missing for predicted output. It took me a while to figure out that I was reading the first row as the row header. Fixing that improved the score (0.50000) of my model. I also faced other issues due to being a first time Kaggler. I cleaned up the MailOrder_Test dataset and in the process some rows got deleted, which is a big no-no. Other issues faced include using a validation set accuracy target without setting a validation dataset, not resetting the index after dropping rows/columns etc.

After the initial solution was done using basic Linear Regression, I moved on to using hyperparameter tuning. This improved the score (0.50154), although still pretty low.

I moved on to XGBoost algorithm to get a better score. This worked fine and I got a better score (0.72071) using XGBoost with minor hyperparameter tuning. However, I wanted to get a better score and ended up cleaning the dataset even further by removing low variance columns & doing a more expansive hyperparameter tuning. This created my best score of 0.79216.

IV. Results

Model Evaluation and Validation

- **Customer Segmentation**

Fig 17: Azdias vs Customers (% of total) per cluster

Some of the provided data is not defined and some are not well understood. Since this is just the centroid for the clusters the actual results will not match exactly but have a small delta difference. Additionally, some of the attributes may not be in agreement with each other. Mostly there is a sort of loose definition for what group of people make up a higher percentage of customers and what group of people do not.

I looked at cluster 9 where the % of customers were much larger than the % of general population. I also looked at cluster 6 where % of customers were much smaller than the % of general population.

People in cluster 9, with high % of customers, tend to have the following characteristics:

- Older, smaller family
    - Smaller family
    - Mature couple
    - No children
    - Relatively older (46-60 years)
    - Golden ager
    - Low mobility
    - Have pet(s)
    - Takes care of themselves (Luxury clothing, education, gardening, food, dietary supplements, wine, medicine, leisure, travel, shoes etc)
    - 3 HH/Building
- Financially prosperous
    - High earner
    - Average investor
    - Average money saver
    - Consumption oriented
- Mail Order history:
    - Has quite a bit of MO history
    - Double buy (12 months) of further mail order
    - Actuality of last transaction for MO total is high
    - Actuality of last transaction for MO online increased
    - % of online transactions within all in MO is 80%
- Cars
    - Avg cars
    - Not much of a difference than the other cluster
    - High % of car in HH
    - Low % of upper- & middle-class cars
- Shopping habits
    - Advertisement interested online shopper
    - Gourmet shopper

    - Multi and double buys of luxury clothing, education, gardening, food, dietary supplements, wine, medicine, leisure, travel, shoes and the like
- Social
    - Average affinity for: Family, Materials, Tradition, Religion, Rational, Tradition, Lust

People in cluster 6, with low % of customers, tend to have the following characteristics:

- Young family or single
    - HH size of ~1
    - Young couple
    - Relatively younger (80's, 30-45 years)
    - Homeland connected vacationist
    - 6 HH/Building
- Financially comfortable
    - Active Middle Class
    - Comfortable
    - Low income
    - Low investment earner
    - Low money saver
- Mail Order history
    - Not much of a history with Mail Orders
- Cars
    - Average cars
    - Not that much of a difference than the other cluster
    - Average % of car in HH
    - Very low % of upper- & middle-class cars
- Shopping habits
    - Advertisement interested online shopper
    - Stressed shopper
    - Multi buys of Books/CDs, technology, mobile, further clothing
- Social

- Very low affinity for: Religion
- Low affinity for: Traditions, family, rational, material and culture
- Average affinity for: Social
- Very high affinity for: Lust

To summarize, Arvato customers seem to be older, financially better, traditional/religious, have pets and some history of mail orders. Non-customers seem to be younger, financially comfortable, less traditional/religious, no pets and very little history of mail orders.

- **Customer Prediction**

  For customer prediction, I worked with two algorithms – Linear Regression (also called Linear Learner in AWS) and XGBoost. Using Linear Regression, I got a best score of 0.50154.

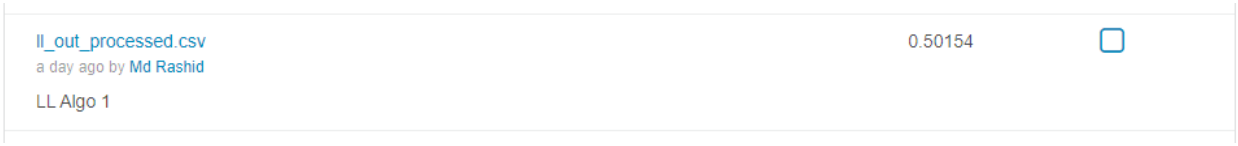


Fig 18: Best score for Linear Learner algorithm on Kaggle

Using XGBoost algorithm, I got a best score of 0.79216 which secured me a position of #106th. This is a huge boost from the previous score. However, even this can be improved with more feature engineering and hyperparameter tuning.

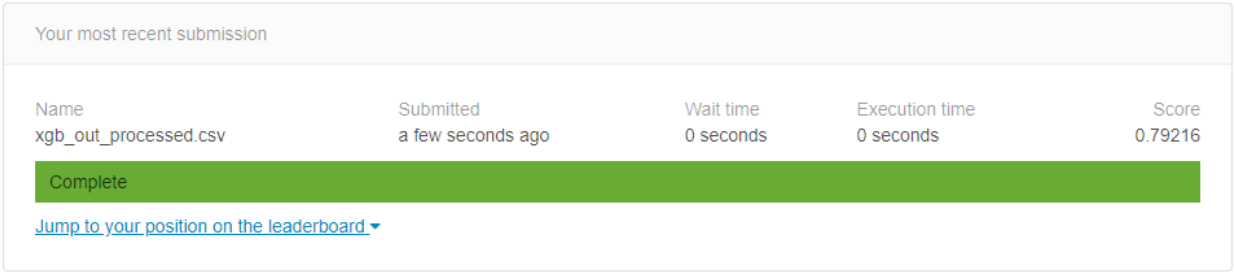


Fig 19: Best score XGBoost algorithm on Kaggle

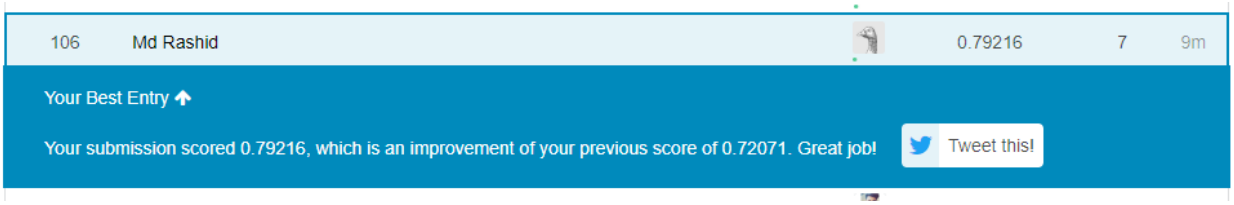


Fig 20: Position on Leaderboard on Kaggle for best score

Justification

Looking at the score of both the algorithms, XGBoost is clearly the better choice here. It improves over Linear Regression score by a margin of almost 30%. My score also puts me in between the two scores that I used as benchmark.

V. Conclusion

Reflection

The whole process can be summarized as:

- Loading data
- Processing data (cleaning, reducing dimensionality by using PCA)

- Apply K-Means algorithm to divide the data into clusters to better understand and compare general population vs customers.
- Create Linear Regression and XGBoost algorithms to predict whether someone is a good match for targeted advertisement.

Improvements

This project was done in rather haste since the deadline to renew subscription for another month was close. Otherwise, some interesting improvements might have been done:

- More feature engineering
    - It may have been possible to drop fewer data rows, specially for positive responses (in MailOut_Train dataset)
- Work with more algorithms to get a higher score
    - I was planning to work with other algorithms like CNN etc
- Work with more hyperparameter tuning to improve score
    - I did some limited work with hyperparameter tuning due to time and budget constraints.
- More optimized and organized code
    - I am relatively new to python. I tried to do my best to keep the code organized. It is possible to optimize and improve the code further.

References

1. https://builtin.com/data-science/step-step-explanation-principal-component-analysis
2. https://en.wikipedia.org/wiki/Principal_component_analysis
3. https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
4. http://www.nlpca.org/pca_principal_component_analysis.html
5. https://heartbeat.fritz.ai/understanding-the-mathematics-behind-k-means-clustering-40e1d55e2f4c
6. Robert D. McFadden (14 January 2019). "Lester Wunderman, Father of Direct Marketing, Dies at 98". The New York Times.
7. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
8. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
9. https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a
10. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d#:~:text=XGBoost%20is%20a%20decision%2Dtree,all%20other%20algorithms%20or%20frameworks.
11. https://www.youtube.com/watch?v=OtD8wVaFm6E
12. https://www.kdnuggets.com/2017/10/xgboost-concise-technical-overview.html