Section 1: Project Definition

Project Overview:

In this project, we will analyze demographics data for customers of a mail-order sales company in Germany, comparing it against demographics information for the general population. we'll use unsupervised learning techniques to perform customer segmentation, identifying the parts of the population that best describe the core customer base of the company. Then, we'll apply what we've learned on a third dataset with demographics information for targets of a marketing campaign for the company, and use a model to predict which individuals are most likely to convert into becoming customers for the company. The data that we will use has been provided by Udacity partners at Bertelsmann Arvato Analytics, and represents a real-life data science task.

There are four data files associated with this project:

- `Udacity_AZDIAS_052018.csv`: Demographics data for the general population of Germany; 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).
- `Udacity_MAILOUT_052018_TEST.csv`: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).

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

The "CUSTOMERS" file contains three extra columns ('CUSTOMER_GROUP', 'ONLINE_PURCHASE', and 'PRODUCT_GROUP'), which provide broad information about the customers depicted in the file. The original "MAILOUT" file included one additional column, "RESPONSE", which indicated whether or not each recipient became a customer of the company. For the "TRAIN" subset, this column has been retained, but in the "TEST" subset it has been removed; it is against that withheld column that our final predictions will be assessed in the Kaggle competition.

Otherwise, all of the remaining columns are the same between the three data files. For more information about the columns depicted in the files, you can refer to two Excel spreadsheets provided in the repo. [One of them](./DIAS Information Levels - Attributes 2017.xlsx) is a top-level list of attributes and descriptions, organized by informational category. [The other](./DIAS Attributes - Values 2017.xlsx) is a detailed mapping of data values for each feature in alphabetical order.

Problem Statement:

We will use the information from the first two files to figure out how customers ("CUSTOMERS") are similar to or differ from the general population at large ("AZDIAS"), then use our analysis to make predictions on the other two files ("MAILOUT"), predicting which recipients are most likely to become a customer for the mail-order company.

Metrics:

An AUC-ROC curve from predicted probabilities will be used to evaluate the performance of the classification models. AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AU C, the better the model is at predicting 0 classes as 0 and 1 classes as 1. By analogy, the Higher the AUC, the better the model is at distinguishing between classes. Details can be read from [this](#).

Section 2: Analysis

Data Exploration/Visualization

I have started with azdias(german population data) because it is the largest and contains nearly all features.

```
azdias.describe()
```

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTE
count	8.912210e+05	891221.000000	817722.000000	817722.000000	81058.000000	29499.000000	6170.000000	120
mean	6.372630e+05	-0.358435	4.421928	10.864126	11.745392	13.402658	14.476013	1
std	2.572735e+05	1.198724	3.638805	7.639683	4.097660	3.243300	2.712427	
min	1.916530e+05	-1.000000	1.000000	0.000000	2.000000	2.000000	4.000000	
25%	4.144580e+05	-1.000000	1.000000	0.000000	8.000000	11.000000	13.000000	1
50%	6.372630e+05	-1.000000	3.000000	13.000000	12.000000	14.000000	15.000000	1
75%	8.600680e+05	-1.000000	9.000000	17.000000	15.000000	16.000000	17.000000	1
max	1.082873e+06	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	1

As it can be seen there are so many features to analyze. To gain some time, we may want to eliminate less important features while looking at missing ratio. Higher missing ratio leads to low information gain so may consider to drop these features.

Let's look at missing value distribution in terms of row basis. We have 93 feature with completely filled values. So I have checked the excel file, it seems that there are features with special values which means that some of missing values are assigned to spesific value. So we need to do conversion to calculate missing ratios correctly.

```
values[values['Attribute']=='AGER_TYP']
```

	Attribute	Description	Value	Meaning
0	AGER_TYP	best-ager typology	-1	unknown
1	AGER_TYP	best-ager typology	0	no classification possible
2	AGER_TYP	best-ager typology	1	passive elderly
3	AGER_TYP	best-ager typology	2	cultural elderly
4	AGER_TYP	best-ager typology	3	experience-driven elderly

As it can be seen that the dataset has special values to indicate the missing values. Therefore, we need to convert values to null to evaluate missing values correctly. I have prepared a script that fixes that issue.

```
df_missing_new.sort_values(by='missing_ratio', ascending=Fa
```

	column_names	n_missing	missing_ratio
7	ALTER_KIND4	890016	0.998648
6	ALTER_KIND3	885051	0.993077
5	ALTER_KIND2	861722	0.966900
4	ALTER_KIND1	810163	0.909048
1	AGER_TYP	677503	0.760196
100	EXTSEL992	654153	0.733996
300	KK_KUNDENTYP	584612	0.655967
8	ALTERSKATEGORIE_FEIN	262947	0.295041
61	D19_LETZTER_KAUF_BRANCHE	257113	0.288495
53	D19_GESAMT_ONLINE_QUOTE_12	257113	0.288495
69	D19_SOZIALES	257113	0.288495
62	D19_LOTTO	257113	0.288495
57	D19_KONSUMTYP	257113	0.288495
85	D19_VERSAND_ONLINE_QUOTE_12	257113	0.288495
77	D19_TELKO_ONLINE_QUOTE_12	257113	0.288495
92	D19_VERSI_ONLINE_QUOTE_12	257113	0.288495
36	D19_BANKEN_ONLINE_QUOTE_12	257113	0.288495
134	KBA05_DIESEL	133324	0.149597
136	KBA05_GBZ	133324	0.149597
135	KBA05_FRAU	133324	0.149597

After conversion, it seems that we don't have that many missing columns. We can drop if they are higher than %20. So, we will drop 17 attributes. Still, we have so many missing features. We may want to drop highly cardinal variables. Because, most of ML algorithm requires one-hot encoding to use categorical variables. So that high cardinal variable creates issue in that sense. The following variables assigned as categorical; actions are also shown in the below.

CAMEO_DEUG_2015: CAMEO classification 2015 - Upper group. It is like some kind of classification we may want to keep it.

CAMEO_DEU_2015: CAMEO classification 2015 - detailed classification. We can drop it because we already have upper group segmentation

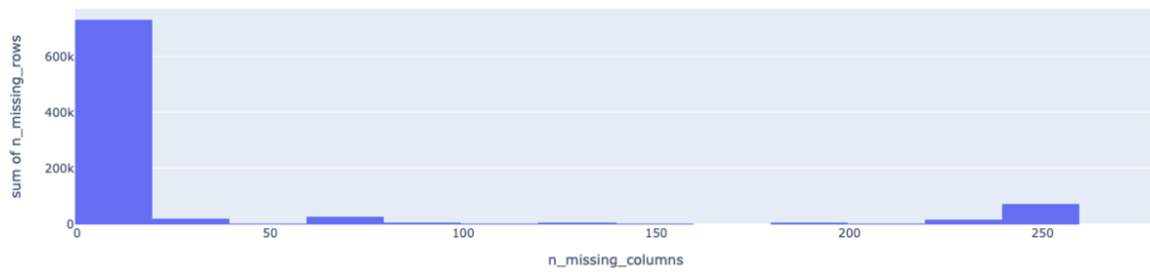
CAMEO_INTL_2015: we don't' have info about this feature we can drop it.

D19_LETZTER_KAUF_BRANCHE: we don't' have info about this feature we can drop it.

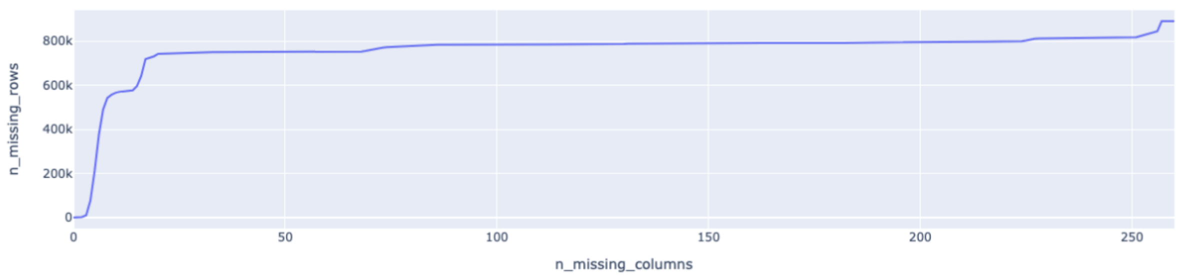
EINGEFUEGT_AM we have so many features so we can drop time related attributes.

OST_WEST_KZ: flag indicating the former GDR/FRG. We can convert it to numerical attribute

Apart from that, we have 4 data frames, so we need to drop non-common features. After dropping all necessary columns, we can do missing row investigation for each row. AS a result, we can check whether we need to drop observations due to high missing columns.



It seems that majority of the population have 0-19 missing columns. To be sure on this let's do it by cumulatively.



After 21, trend normalizes so we can choose 21 for missing rows elimination.

Section 3: Methodology

Data Preprocessing:

There are 2 different problems in our cases first one is identifying how much companies' customer segments similar to the German population which is unsupervised methodology because we don't know the labels. Second one is that selected groups are going to response to our campaigns which is supervised because we have already labeled data. For unsupervised learning, we have used k-means clustering algorithm and PCA to decrease the size of the features. Clustering algorithms are very sensitive to the data, so we needed to do missing imputations, outlier elimination, standardizations and dimension reduction. We have decided to following transformations for each data types.
Binary: we need to determine 2 value features. We can assign missing values as most popular value.
Categorical: we need to do one-hot encoding to convert categorical to binary features
Numerical: we can use median value to impute missing values to overcome outliers' effects.
For the supervised learning side, I have only applied XGBoost model due to time constraint and memory issues. We have invested time on hyperparameter tunning. For the XGBoost there is

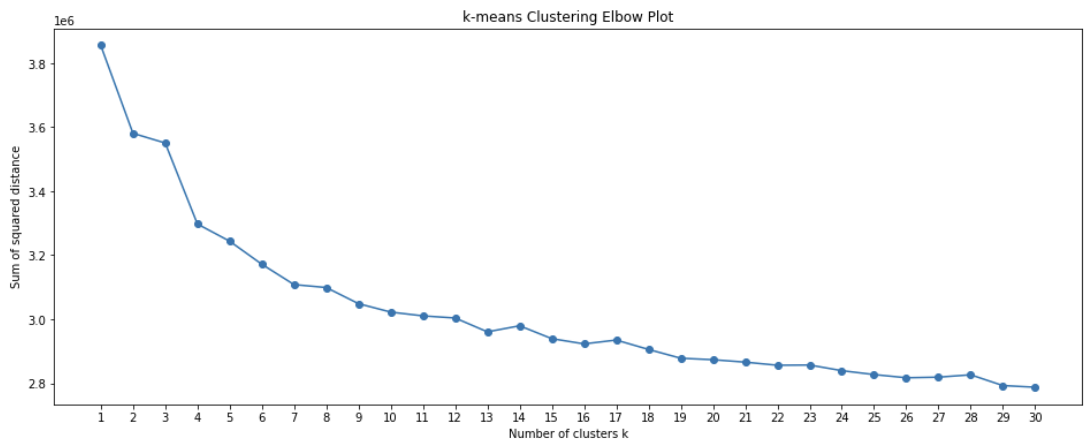
no need for feature transformation because decision tree algorithms handle all missing values, outliers etc.

Implementation:

While doing transformation for clustering algorithm, we have faced with memory issue because we have many features. So, we have decided to drop correlated features which means that they have similar meaning. Looking at correlation also took so long but it is important that we did that only once however for transformation or PCA or other algorithms we had needed to deal with these problem multiple times. Therefore, dropping correlated attributes improved our performance on the overall. In addition to that, we had to random resampling while doing clustering because again we hit the memory issue.

Refinement:

As we have stated before, we have used sklearn's KMeans clustering algorithm to identify different segments of the data. In KMeans clustering we had to choose the number of clusters that the algorithm will find. This can be difficult since we don't know how many clusters exist. So we choose number of clusters with elbow method.



There was no certain elbow in the graph however it seemed that there is slow decrease after 13th cluster.

For the XGBoost algorithm, we have split train dataset as test and train then used GridSearchCV to automate the tuning of hyperparameters. Also, we have unbalanced dataset. For imbalanced dataset, we needed to arrange scale_pos_weight parameter. It is documented that it is better to use if there is imbalanced data.

scale_pos_weight = total_negative_examples / total_positive_examples

As a result, the following parameters are chosen as best.

{'gamma': 1.0, 'learning_rate': 0.05, 'max_depth': 2, 'reg_lambda': 0, 'scale_pos_weight': 80}

Section 4: Results

Model Evaluation, Validation, Justification:

As a result, we have found out 13 clusters like in the following.

It seems that 1st group targeted most. 11th, 5th and 10th clusters follow it accordingly. 9th cluster customers are outside of the company focus group because they are dominating population but not in companies customer segment. 12th and 13th groups follow it accordingly.

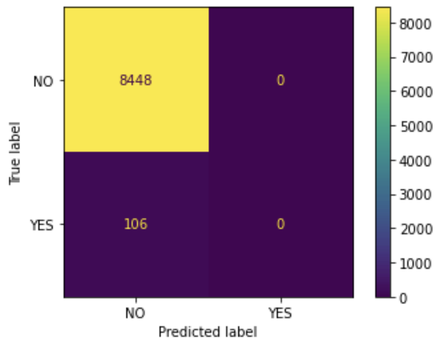
When we look at clusters' details to understand the groups behavior, the following tabular data show us features values on specified clusters.

	Attribute	interested_cluster_1	interested_cluster_11	not_interested_cluster_9	not_interested_cluster_12	Description	Value	Meaning
0	LNR	655026.752720	652236.260177	649125.931917	656533.384413	NaN	NaN	NaN
1	EINGEZOGENAM_HH_JAHR	2005.234318	2004.746256	2002.984422	1998.153626	NaN	NaN	NaN
2	ANZ_HAUSHALTE_AKTIV	6.013276	10.585922	6.216955	7.297482	number of households in the building	...	numeric value (typically coded from 1-10)
3	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	0	unknown / no main age detectable
4	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	1	01.01.1895 bis 31.12.1899
5	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	2	01.01.1900 bis 31.12.1904
6	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	3	01.01.1905 bis 31.12.1909
7	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	4	01.01.1910 bis 31.12.1914
8	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	5	01.01.1915 bis 31.12.1919
9	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	6	01.01.1920 bis 31.12.1924
10	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	7	01.01.1925 bis 31.12.1929
11	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	8	01.01.1930 bis 31.12.1934
12	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	9	01.01.1935 bis 31.12.1939
13	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	10	01.01.1940 bis 31.12.1944
14	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	11	01.01.1945 bis 31.12.1949
15	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	12	01.01.1950 bis 31.12.1954
16	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	13	01.01.1955 bis 31.12.1959
17	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	14	01.01.1960 bis 31.12.1964
18	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	15	01.01.1965 bis 31.12.1969
19	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	16	01.01.1970 bis 31.12.1974
20	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	17	01.01.1975 bis 31.12.1979
21	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	18	01.01.1980 bis 31.12.1984
22	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	19	01.01.1985 bis 31.12.1989
23	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	20	01.01.1990 bis 31.12.1994
24	ALTER_HH	10.632381	13.574433	9.114098	12.818555	main age within the household	21	01.01.1995 bis 31.12.1999
25	GEBURTSJAHR	679.955966	595.566755	2123.094327	217.495125	year of birth	...	numeric value
26	KBA13_ANZAHL_PKW	631.058996	687.242565	467.580051	786.378252	number of cars in the PLZ8	...	numeric value

As a result, clusters are very close to each other so to be able to interpret correctly we can focus on most focused and unfocussed groups which are cluster 1 and cluster 9.It seems that marketing team mostly focus on younger groups and who have cars.

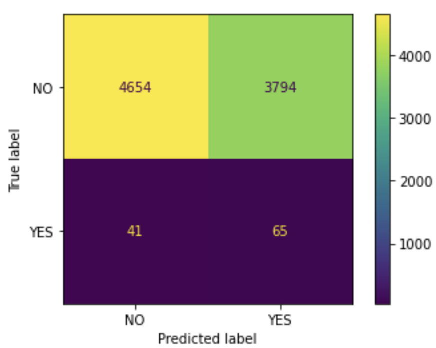
For the supervised learning, we have firstly developed preliminary XGBoost model, however the performance metrics were really bad like the following.

```
Classification report (Test):

              precision    recall  f1-score   support

           0       0.99      1.00      0.99      8448
           1       0.00      0.00      0.00       106

    accuracy                           0.99      8554
   macro avg       0.49      0.50      0.50      8554
weighted avg       0.98      0.99      0.98      8554


Train Accuracy: 0.9876071706936866
Test Accuracy: 0.9876081365443068

Train Recall: 0.0
Test Recall: 0.0
```



It seems that the model tends to say customer will not answer. So we have played around parameters a little bit. The biggest reason might be having unbalanced dataset. So to overcome this, we have calculated scale_pos_weight as 80.

```
Classification report (Test):

              precision    recall  f1-score   support

           0       0.99      0.55      0.71      8448
           1       0.02      0.61      0.03       106

    accuracy                           0.55      8554
   macro avg       0.50      0.58      0.37      8554
weighted avg       0.98      0.55      0.70      8554


Train Accuracy: 0.5563133281371785
Test Accuracy: 0.5516717325227963

Train Recall: 0.720125786163522
Test Recall: 0.6132075471698113

Confusion matrix (Test):

57]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fe88f2cc9a0>
```



Final result is much better. However still we don’t have well performing model. We have discussed what can be done to improve model in the Conclusion section.

Section 5: Conclusion

My biggest challenge was related about size of the data. I had to restart kernel multiple times. To overcome this problem, unfortunately I had to drop correlated features mostly they were personal data which would help me get interesting clusters. Without them, I came across with similar clusters. The maximum result I could get from clustering was that marketing team mostly focus on younger groups and who have cars. For the supervised learning side there are plenty of room to improve such as trying other classification algorithms like Gradient Boost, Ada boost, SVM, etc. Apart from that, we can try to segment the data and fit supervised algorithm accordingly. This might be another solution to develop specific solutions to specific customer segments.