

---

# **Customer Segmentation & Prediction**

**Arvato Bertelsmann**

## **PROJECT REPORT**

### **CAPSTONE PROJECT**

---

**UDACITY MACHINE LEARNING NANO DEGREE**

**Authored by: Ambresh Patil**



---

# Project Overview

This project is a part of Udacity Machine Learning Engineer Nanodegree Capstone Project in collaboration with Bertelsmann Arvato. Demographic information of customers of a mail order company dealing with organic products was analyzed with demographic information of general population of Germany to identify customer segments and data of mail order marketing campaign was used to predict customer conversion. The mail order company is a client of Bertelsmann Arvato. The datasets were provided by Arvato. The data is protected by terms and conditions by Arvato and is not available for use by general public.

Customer Segmentation was performed using Unsupervised learning to identify the parts of the population that best describes the core customer base of the company. With above analysis, Supervised learning was applied on marketing campaign data to predict which individuals are likely respond or becoming customer. Performance scoring was done by uploading the prediction to Kaggle Competition.

The data provided by Arvato is as follows

- **Azdias** — Demographics data for the general population of Germany (891 211 x 366).
- **Customers** — Demographics data for customers of the German mail-order company (191 652 x 369).
- **Mailout\_train (train)** — Demographics data for individuals who were targets of a marketing campaign (42 982 x 367).
- **Mailout\_train (test)** — Demographics data for individuals who were targets of a marketing campaign (42 833 x 366).
- **DIAS Attributes — Values 2017.xlsx** — a data dictionary of the columns including description, possible values, and what the values mean.
- **DIAS Information Levels — Attributes 2017.xlsx** — indicates the information level for each column (Person, Household, Building, etc)

**feature\_summary\_complete.csv** was manually created using two DIAS description files. This file contains attribute, information level, data type, missing or unknown values.

---

# Problem Statement

How can the efficiency of customer acquisition be increased of a client who is into selling of organic product and intends to acquire new customer by sending sample products to customer segment of interest?

My proposal to address problem statement

1. An unsupervised learning approach to identify potential customer segment by analyzing attributes of existing customer against the general population data.
2. Building a Supervised learning model based on the former analysis, which predicts if an individual will respond to the campaign or no. The dataset used for prediction will be targets of client's mail order campaign.
3. The chosen model will be used to make prediction and will be scored on campaign data as a part of Kaggle competition.

## Metrics

As this is a multi-class classification problem, Area Under the Curve Receiver Operating Characteristics (ROC-AUC) is the key metrics to measure model performance.

The curve represents a measure of separability and, the higher the score the better the performance of the model.

ROC-AUC provides immunity to class imbalance, which is crucial for this problem.

Typically, we see number of positive responders to an ad campaign are far lesser than negative responders.

This is also the required evaluation metric for the Kaggle submission.

# Analysis

## Data Exploration

There are two data description Excel spreadsheets and four data files associated with this project:

- **AZDIAS**: Demographics data for the general population of Germany. It has 891211 persons (rows) and 366 features (columns).

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN
count	8.912210e+05	891221.000000	817722.000000	817722.000000	81058.000000	29499.000000	6170.000000	1205.000000	628274.000000
mean	6.372630e+05	-0.358435	4.421928	10.864126	11.745392	13.402658	14.476013	15.089627	13.700717
std	2.572735e+05	1.198724	3.638805	7.639683	4.097660	3.243300	2.712427	2.452932	5.079849
min	1.916530e+05	-1.000000	1.000000	0.000000	2.000000	2.000000	4.000000	7.000000	0.000000
25%	4.144580e+05	-1.000000	1.000000	0.000000	8.000000	11.000000	13.000000	14.000000	11.000000
50%	6.372630e+05	-1.000000	3.000000	13.000000	12.000000	14.000000	15.000000	15.000000	14.000000
75%	8.600680e+05	-1.000000	9.000000	17.000000	15.000000	16.000000	17.000000	17.000000	17.000000
max	1.082873e+06	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.000000

- **CUSTOMERS**: Demographics data for customers of a mail-order company. It has 191652 rows and 369 features.

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN
count	191652.000000	191652.000000	145056.000000	145056.000000	11766.000000	5100.000000	1275.000000	236.000000	139810.000000
mean	95826.500000	0.344359	1.747525	11.352009	12.337243	13.672353	14.647059	15.377119	10.331579
std	55325.311233	1.391672	1.966334	6.275026	4.006050	3.243335	2.753787	2.307653	4.134828
min	1.000000	-1.000000	1.000000	0.000000	2.000000	2.000000	5.000000	8.000000	0.000000
25%	47913.750000	-1.000000	1.000000	8.000000	9.000000	11.000000	13.000000	14.000000	9.000000
50%	95826.500000	0.000000	1.000000	11.000000	13.000000	14.000000	15.000000	16.000000	10.000000
75%	143739.250000	2.000000	1.000000	16.000000	16.000000	16.000000	17.000000	17.000000	13.000000
max	191652.000000	3.000000	9.000000	21.000000	18.000000	18.000000	18.000000	18.000000	25.000000

- **MAILOUT\_TRAIN**: Demographics data for individuals who were targets of a marketing campaign; 42982 persons and 367 features including response of people.
- **MAILOUT\_TEST**: Demographics data for individuals who were targets of a marketing campaign; 42833 persons and 366 features.

---

I see a lot of missing values in these datasets and not all the features have explanation in a given Excel spreadsheets which needs to be addressed.

Arvato provided two spreadsheets containing a top-level list of attributes and descriptions, and a detailed mapping of data values for each feature. Feature\_summary\_complete.csv was created using two DIAS description files. This file contains attribute, information level, data type, missing or unknown values.

	attribute	information_level	type	missing_or_unknown
0	AGER_TYP	person	categorical	[-1,0]
1	ALTERSKATEGORIE_GROB	person	ordinal	[-1,0,9]
2	ANREDE_KZ	person	categorical	[-1,0]
3	CJT_GESAMTTYP	person	categorical	[0]
4	FINANZ_MINIMALIST	person	ordinal	[-1]

## Algorithms and Techniques

For the Unsupervised Learning approach section I will be using Principal Component Analysis algorithm for dimensionality reduction. At its core, PCA allows the decomposition and transformation of data into the most relevant component though the principal that the more variance a feature has, the greater is the power of understanding of the data.

After dimensionality reduction I will implement K-means as a clustering approach.

**Customer Segmentation** is the practice of partitioning a customer base into groups of individuals that have similar characteristics, for this purpose K-means is a simple and fast approach that fits well this problem since it scales quite well to large datasets. The principle of K-means implementation is quite elegant:

- Specify number of clusters K (that in this case are obtained through optimization using the elbow method)
- Initialize centroids by first shuffling the dataset and select random K points

- 
- Iterate until the centroids don't change anymore.

It is important to note that in the case of K-means there is no ground truth to evaluate model's performance there is no single right answer to evaluation since for instance the number of k clusters is a hyperparameter input.

Elbow Method is used to optimize the number of clusters that were ideal for this dataset. The concept behind the elbow method is to run the k-means clustering on the dataset on ranges of k while calculating the sum of squared errors for each k tested. Once plotted the line looks like an elbow. The elbow point represents the point of diminishing returns when the k increases.

As we have a well stated base for the segmentation, I will be using Supervised learning approach for prediction. For the Supervised portion of this analysis I decided to use a parallel testing approach and will be testing data with below models for best performance.

XGBClassifier - It is implementation of gradient boosted decision trees. Models are added sequentially and new models are created to calculate prior errors and add the differences to get the final prediction.

LogisticRegression – Uses historical campaign data to create patterns based on customer responses.

RandomForestClassifier - Random forest performs vectorization, building a classifier fitted to the random target vectors and counting observations in the same node. It then measures the distance between observations.

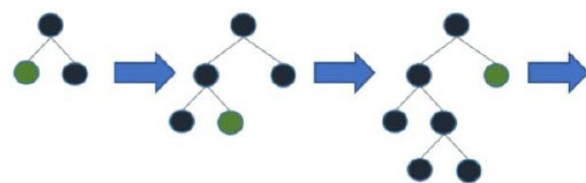
MLPClassifier – A model based on back propagation neural networks. The weights of different neurons are updated in a way that the difference between the desired and predicted output is as small as possible



GradientBoostingClassifier - A Gradient boosting model, similar to XGB but slower, can't be parallelized and does not perform tree regularization like XGB to avoid data overfitting.

LGBMClassifier - Light GBM is a gradient boosting framework that uses tree-based leaf-wise learning algorithm rather than level wise algorithm.

Leaf-Wise Tree Growth



Level-Wise Tree Growth



## Benchmark

To determine how changes in data processing and model hyperparameter tuning were affecting my models I decided to first create base scores of the different models on cleaned but unscaled data. Out of all the models used in this solution LR is the simplest and more straightforward model to use as a benchmark.

Unscaled			StandardScaler		MinMaxScaler	
	Model	Score		Model	Score	
0	LR	0.6660		standardLR	0.6613	minmaxLR 0.6745
1	RF	0.6351		standardRF	0.6376	minmaxRF 0.6352
2	XGB	0.7607		standardXGB	0.7607	minmaxXGB 0.7607
3	LGBM	0.7117		standardLGBM	0.7191	minmaxLGBM 0.7117
4	GB	0.7571		standardGB	0.7571	minmaxGB 0.7571
5	MLP	0.6026		standardMLP	0.5915	minmaxMLP 0.5914

I created a reference table to the non-optimized models to be compared against the LR results.

# Data Preprocessing

While loading the data I encountered warning of mixed data type and an unnamed column

```
/home/ec2-user/anaconda3/envs/mxnet_p36/lib/python3.6/site-packages/IPython/core/interactiveshell.py:2785: DtypeWarning: Columns (19,20) have mixed types. Specify dtype option on import or set low_memory=False.
interactivity=interactivity, compiler=compiler, result=result)
```

The features showing mixed type error was determined

```
# Investigating column 19 and 20.
print(azdias.iloc[:,19:21].columns)
print(customers.iloc[:,19:21].columns)
```

```
Index(['CAMEO_INTL_2015', 'CJT_GESAMTTYP'], dtype='object')
Index(['CAMEO_INTL_2015', 'CJT_GESAMTTYP'], dtype='object')
```

The columns were a mixture of strings, floats and a missing values marker ['X'] or ['XX'], A function was created to convert the strings to floats and the 'X' marker to np.nan.

Customers data had 3 columns which were not present in Azdias data hence I dropped them to maintain the homogeneity of features between these two datasets which will be used for segmentation. This brings both datasets to 366 features

## Azdias Shape ¶

```
# checking how the azdias dataframe looks like
print('Printing dataframe shape')
print(azdias.shape)
print('_____')

azdias.head()
```

```
Printing dataframe shape
(891221, 366)
```



---

## Customers Shape

```
# checking how the customer dataframe looks like
print('Printing dataframe shape')
print(customers.shape)
print('_____')

customers.head()
```

```
Printing dataframe shape
(191652, 366)
```

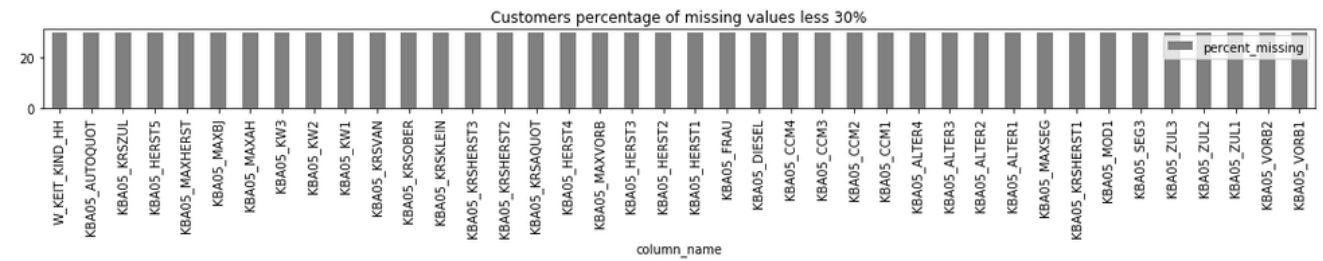
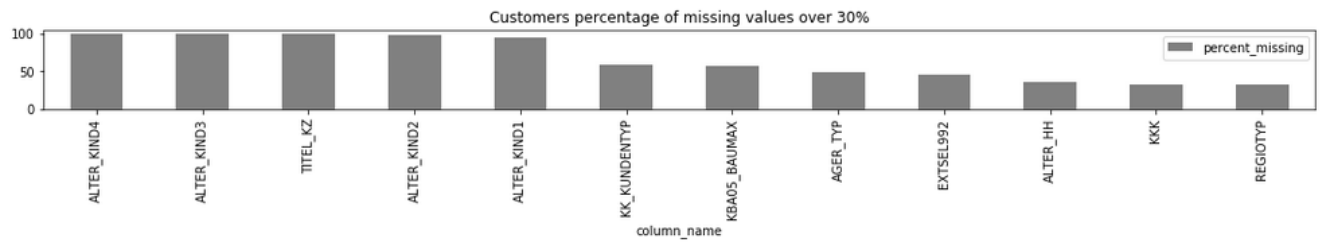
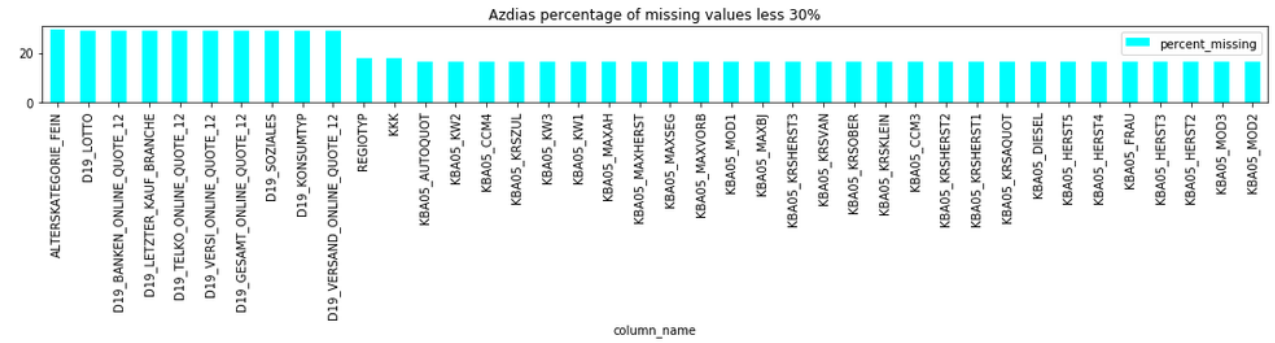
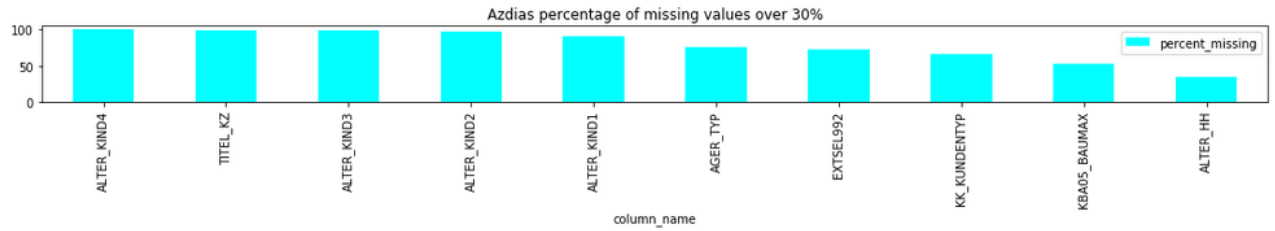
## Dealing with missing values

A function is created that identifies missing and unknown data based on the information provided in the DIAs file and replaces these tags with NaNs.

The results after running the missing function is

```
Identified missing data in Azdias:
Pre-cleanup: 33493669 Post_cleanup: 37088263
Identified missing data in Customers:
Pre-cleanup: 13864774 Post_cleanup: 14488721
```

I determined the threshold for missing data columns to be dropped be 30%. Any columns which are missing over 30% of data be dropped.



---

An inconsistency between the azdias and the customers data is created after dropping columns which are missing more than 30% data

```
#since I just dropped several columns I will do another balance check
balance_checker(azdias, customers)

Feature balance between dfs?: False
Your first argument df differs from the second on the following columns:
{'KKK', 'REGIOTYP'}
Your second argument df differs from the first on the following columns:
set()
```

Hence these 2 columns are dropped from azdias.

When I considered to delete rows where the data is missing, the performance was affected, hence no rows were dropped.

## Feature Engineering

Below are some Feature Encoding and Engineering performed.

- 'ANREDE\_KZ' refers to lifestyle characteristics and used One Hot Encoding to encode it.
- 'EINGEFUEGT\_AM' is a time related feature, so converted it into a datetime object and extracted the year.
- 'CAMEO\_DEU\_2015' is a categorical feature that ranged from 1 to 9 and A to F I used a LabelEncoder() to encode the categories to ints.
- 'OST\_WEST\_KZ' is a binary feature that had the values array ['W', 'O'], which I mapped to: {'W':0, 'O':1}
- I created 2 different features from 'PRAEGENDE\_JUGENDJAHRE', a 'DECADE' feature and a type of 'MOVEMENT' feature (avant-garde or not)
- 'WOHNLAG' refers to neighborhood area, from very good to poor; rural so I created 2 different features from this one 'QUALITY' related to the quality of the borough and 'AREA' to identify if it is rural or not
- 'LP\_LEBENSPHASE\_FEIN' was used to create two new features, one related to life stage, 'LP\_LEBENSPHASE\_FEIN\_life\_stage' and one related to the wealth scale 'LP\_LEBENSPHASE\_FEIN\_fine\_scale'

---

SimpleImputer() was used to impute the nans with the most-frequent values.

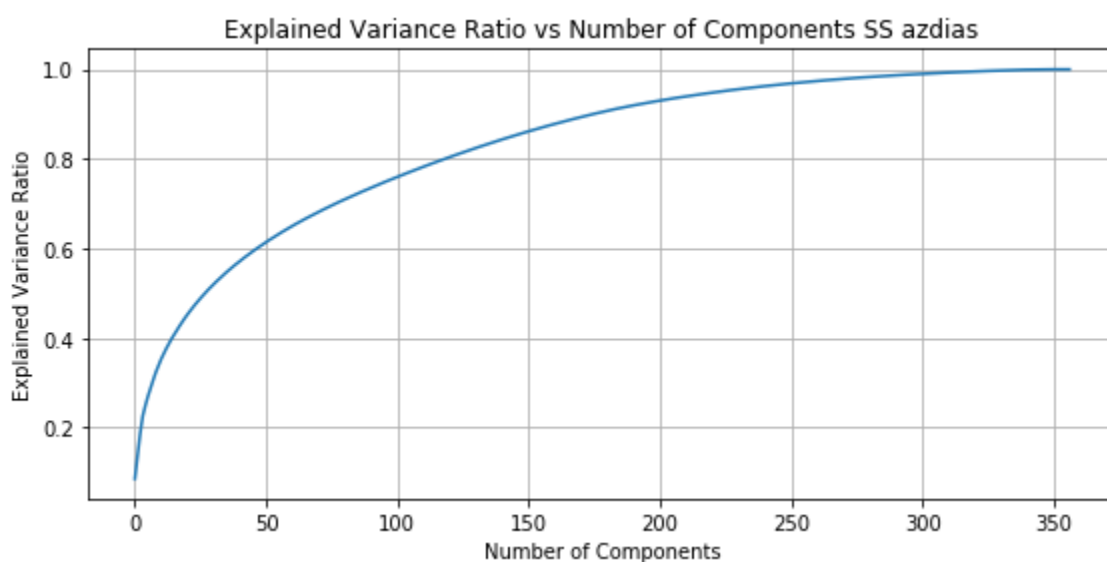
## Dimensionality Reduction using PCA

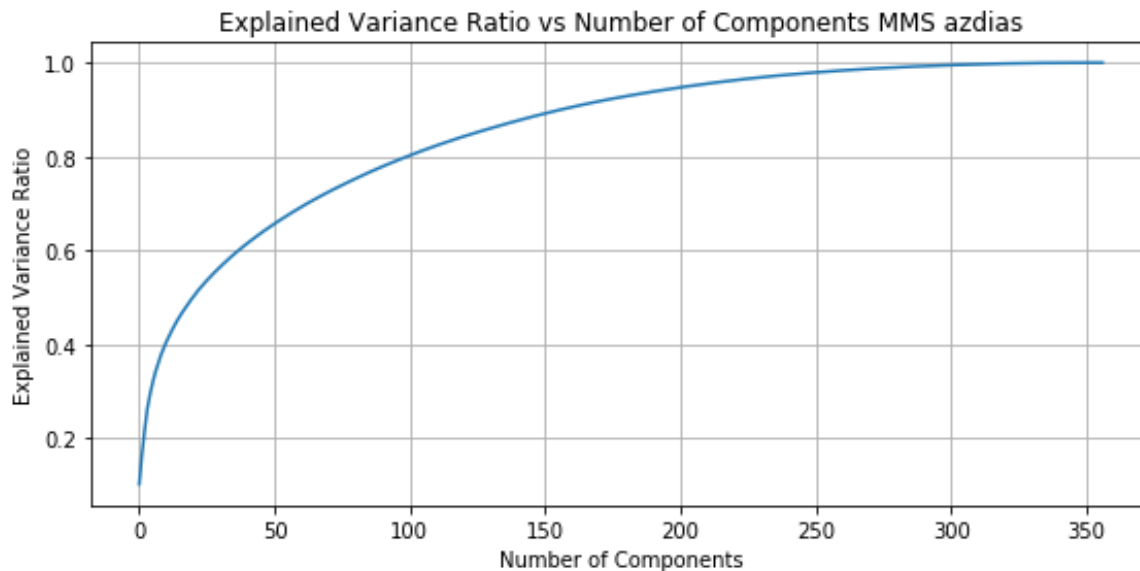
As number of features or dimensions increases, the model becomes more complex, and chances of overfitting increases. Typically, a model trained on large number of features produces prediction which is much more dependent on the data it was trained on leading to over fitting. Hence poor performance on general data. It is important to reduce dimensionality to significantly increase efficiency in computing and memory consumption of a model, to increase performance and reduce feature noise.

The main linear technique for dimensionality reduction, principal component analysis or PCA, performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized.

For scaling the data was tested with Min Max Scaler, Standard Scaler and Robust Scaler and the performances of the standard scaler and min max scaler were identical.

PCA or Principal Component Analysis was performed for dimensionality reduction using feature extraction. Scree plot is used to decide on how many components, that accounted for over 80% of the variance observed





Relevance is directly proportional to weight of the attribute, let's look at the most important features for a few dimensions considering that positive weights might relate to a positive relationship and negative weights a negative one:

dimension 1(0) using standard scaler:

These are some of features related to positive weights:

- o MOBI\_RASTER: refers to the individual's mobility
- o KBA13\_ANTG1: lower share of car owners
- o PLZ8\_ANTG1: lower number of 1-2 family houses

And these are some of the features related to negative weights:

- o CAMEO\_DEU\_2015: detailed classification of cultural and living status
- o KBA13\_ANTG3: refers to possession of higher number of cars
- o PLZ8-ANTG3: number of 6-10 family houses in the PLZ8

The first dimension refers to the social status and living conditions of the individuals.

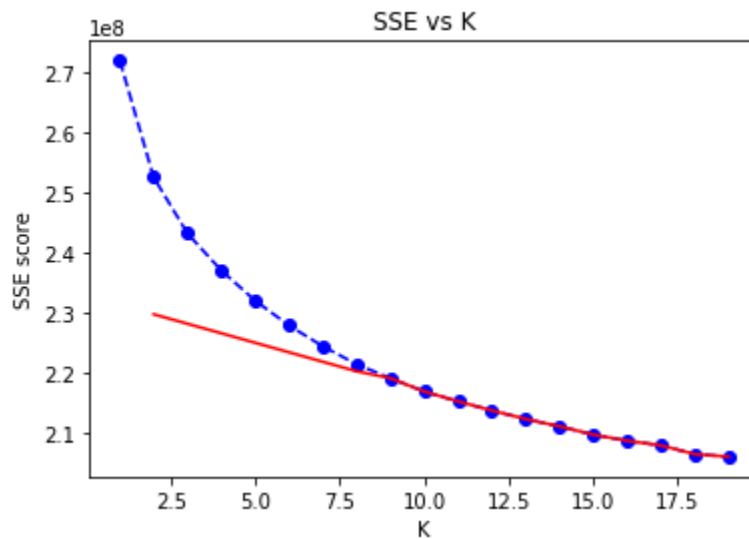
From the scree-plots:

- With Standard Scaler having 150 principal components 90% of the variance can be represented.
- With Minmax Scaler having 150 principal components 90% of the variance can be represented.

## Customer Segmentation - Unsupervised Learning Approach

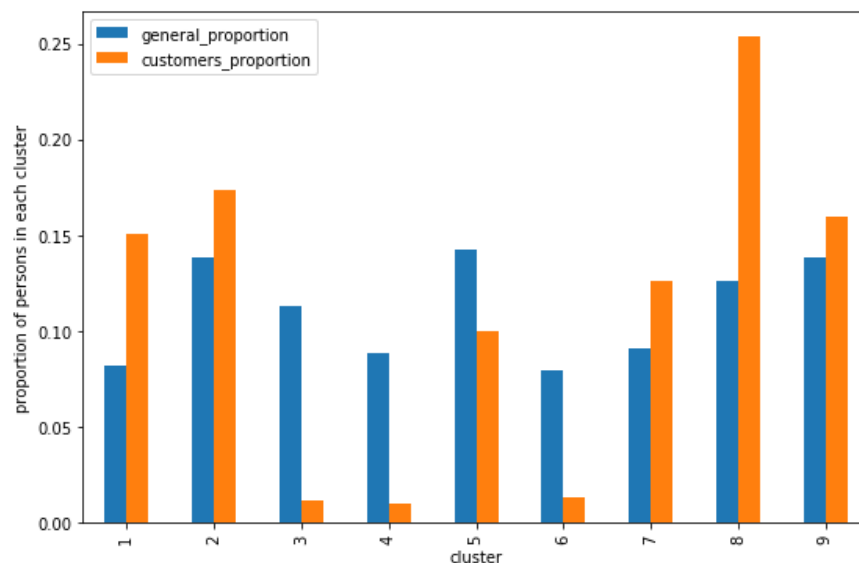
I have chosen to keep 150 components as per PCA after scaling the data using standard scalar. To create clusters of similar data, K-means is used.

We have to define number of centroids, that is k for Kmeans. I will be using Elbow method to find the best value for k. The Elbow method compares the sum of squares until there is no more significant improvement. This k value can be seen in graph where the graph decent becomes gradual.



The above graph hints at the optimal value for k being 9.

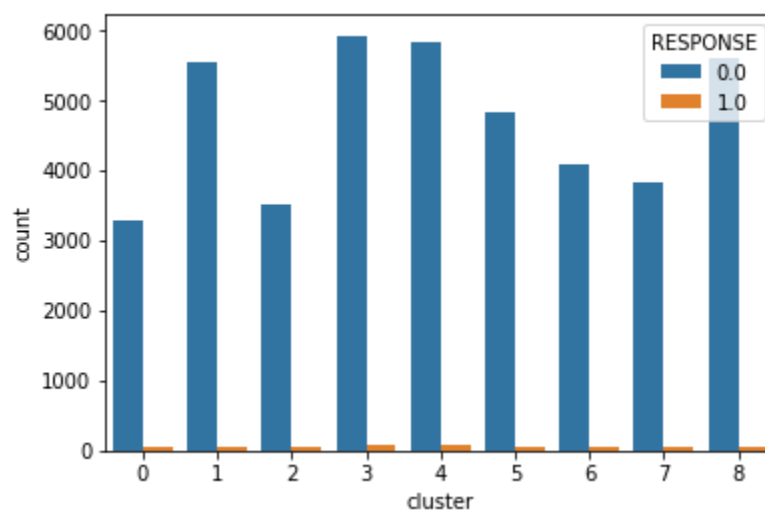
With K means fitted to 9 clusters, I can see some of the clusters which are more in proportion to customer compared to general population.



Using 9 clusters for my analysis I find that cluster 1 and cluster 8 seem to have the features that coin what a core customer is and cluster 3, 4 and 6 round up what excludes who the core customers are.

## Prediction - Supervised Learning Approach

When positive vs negative responses is observed, it shows the imbalance in data





---

Few of the classifier I wanted to try for this project:

**XGBClassifier** - It is implementation of gradient boosted decision trees. Models are added sequentially and new models are created to calculate prior errors and add them up for final prediction.

**LogisticRegression** – Uses historical campaign data to create patterns based on customer responses.

**LGBMClassifier** - Light GBM is a gradient boosting framework that uses tree-based leaf-wise learning algorithm.

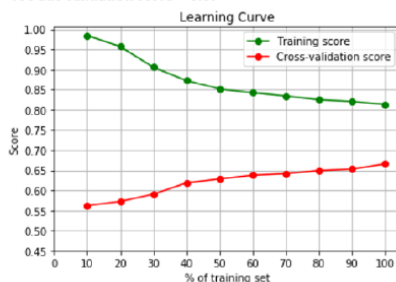
**RandomForestClassifier** - Random forest performs vectorization, building a classifier fitted to the random target vectors and counting observations in the same node. It then measures the distance between observations.

**MLPClassifier** – A back propagation neural network based model.

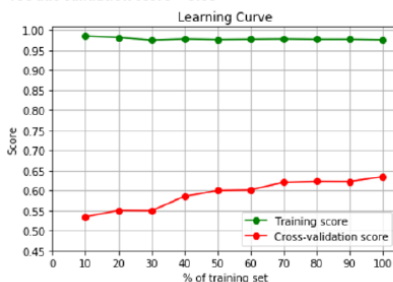
**GradientBoostingClassifier** - A Gradient boosting model.

Lets see below how they perform against each other

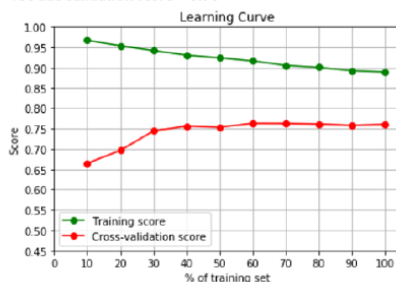
**Linear Regression: 0.665969 (0.010033)**  
 roc auc train score = 0.81  
 roc auc validation score = 0.67



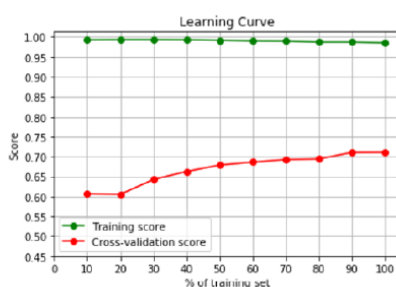
**RandomForestClassifier: 0.635064 (0.022029)**  
 roc auc train score = 0.98  
 roc auc validation score = 0.63



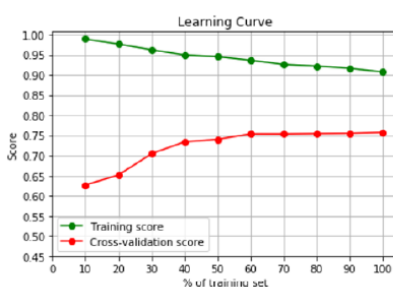
**XGBClassifier: 0.760671 (0.013216)**  
 roc auc train score = 0.89  
 roc auc validation score = 0.76



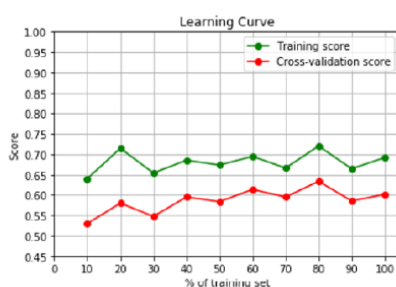
**LGBM: 0.711664 (0.012695)**  
 roc auc train score = 0.99  
 roc auc validation score = 0.71



**GB: 0.757078 (0.005977)**  
 roc auc train score = 0.91  
 roc auc validation score = 0.76



**MLP: 0.602555 (0.066299)**  
 roc auc train score = 0.69  
 roc auc validation score = 0.6



	Model	Score
0	LR	0.6660
1	RF	0.6351
2	XGB	0.7607
3	LGBM	0.7117
4	GB	0.7571
5	MLP	0.6026

We can see that 3 models Gradient Boost, LGBM and XGB seems to be performing well for unscaled data.

---

Now let us see the score with scaling the data using Standard Scaler

	Model	Score	Model	Score
0	LR	0.6660	standardLR	0.6613
1	RF	0.6351	standardRF	0.6376
2	XGB	0.7607	standardXGB	0.7607
3	LGBM	0.7117	standardLGBM	0.7191
4	GB	0.7571	standardGB	0.7571
5	MLP	0.6026	standardMLP	0.5915

Let us see the score with scaling the data using Minmax Scaler

	Model	Score	Model	Score	Model	Score
0	LR	0.6660	standardLR	0.6613	minmaxLR	0.6745
1	RF	0.6351	standardRF	0.6376	minmaxRF	0.6352
2	XGB	0.7607	standardXGB	0.7607	minmaxXGB	0.7607
3	LGBM	0.7117	standardLGBM	0.7191	minmaxLGBM	0.7117
4	GB	0.7571	standardGB	0.7571	minmaxGB	0.7571
5	MLP	0.6026	standardMLP	0.5915	minmaxMLP	0.5914

Since both standard scaler and min max scaler perform equally well, I will stick with minmax scaler.

I have selected LGBM and XGB for testing.

For tuning I select Bayesian optimization approach as it is known to be one of the best performing approach.

The process of hyperparameter tuning for LGBM and XGB is below

---

## Hyperparameter tuning lgbm

Using BayesSearch CV with 500 iterations, below is the hyperparameters selected as per results

Model #499

Best ROC-AUC: 0.774

Best params: {'colsample\_bytree': 1.0, 'learning\_rate': 0.0882410095694084, 'max\_bin': 826, 'max\_depth': 2, 'min\_child\_samples': 0, 'min\_data\_in\_leaf': 32, 'n\_estimators': 20, 'num\_leaves': 94, 'reg\_alpha': 1.0, 'reg\_lambda': 1e-09, 'scale\_pos\_weight': 90.0}

Model #500

Best ROC-AUC: 0.774

Best params: {'colsample\_bytree': 1.0, 'learning\_rate': 0.0882410095694084, 'max\_bin': 826, 'max\_depth': 2, 'min\_child\_samples': 0, 'min\_data\_in\_leaf': 32, 'n\_estimators': 20, 'num\_leaves': 94, 'reg\_alpha': 1.0, 'reg\_lambda': 1e-09, 'scale\_pos\_weight': 90.0}

```
LGBMClassifier(application='binary', boosting_type='gbdt', class_weight=None,
                 colsample_bytree=1.0, importance_type='split',
                 learning_rate=0.0882410095694084, max_bin=826, max_depth=2,
                 metric='auc', min_child_samples=0, min_child_weight=0.001,
                 min_data_in_leaf=32, min_split_gain=0.0, n_estimators=20,
                 n_jobs=-1, num_leaves=94, objective=None, random_state=None,
                 reg_alpha=1.0, reg_lambda=1e-09, scale_pos_weight=90.0,
                 silent=True, subsample=1.0, subsample_for_bin=200000,
                 subsample_freq=0, verbose=0)
```

## Hyperparameter tuning xgboost

Using BayesSearch CV with 200 iterations, below is the hyperparameters selected as per results

Model #199

Best ROC-AUC: 0.7718

Best params: {'colsample\_bytree': 0.5578105865941786, 'gamma': 0.07148993933953135, 'learning\_rate': 0.002858444321957567, 'max\_depth': 4, 'min\_child\_weight': 1, 'n\_estimators': 500, 'reg\_alpha': 0.004635659074237927, 'scale\_pos\_weight': 41, 'subsample': 0.522846932344412}

Model #200

Best ROC-AUC: 0.7718

Best params: {'colsample\_bytree': 0.5578105865941786, 'gamma': 0.07148993933953135, 'learning\_rate': 0.002858444321957567, 'max\_depth': 4, 'min\_child\_weight': 1, 'n\_estimators': 500, 'reg\_alpha': 0.004635659074237927, 'scale\_pos\_weight': 41, 'subsample': 0.522846932344412}

---

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=0.5578105865941786,
              eval_metric='auc', gamma=0.07148993933953135,
              learning_rate=0.002858444321957567, max_delta_step=0, max_depth=4,
              min_child_weight=1, missing=None, n_estimators=500, n_jobs=-1,
              nthread=None, objective='binary:logistic', random_state=0,
              reg_alpha=0.004635659074237927, reg_lambda=1, scale_pos_weight=41,
              seed=None, silent=1, subsample=0.522846932344412, verbosity=1)
```

I will be generating predictions using both xgboost model and LGBM model and submitting at Kaggle competition. Based on the Kaggle score I will be deciding my best model for prediction.

# Results







## Kaggle Score

Public Leaderboard Private Leaderboard

This leaderboard is calculated with approximately 30% of the test data.  
The final results will be based on the other 70%, so the final standings may be different.

Raw Data

Refresh

#	Team Name	Notebook	Team Members	Score ?	Entries	Last
1	Ambresh Patil			0.81063	58	3mo
Your Best Entry ↑						
Your submission scored 0.81063, which is an improvement of your previous score of 0.80901. Great job!				<div> Tweet this!</div>		
2	Telmo			0.80936	57	3mo
3	TensorFrozen(Shihao)			0.80819	29	1y
4	Gaurav Ansal			0.80816	27	9mo
5	weft169Aston			0.80811	62	2mo

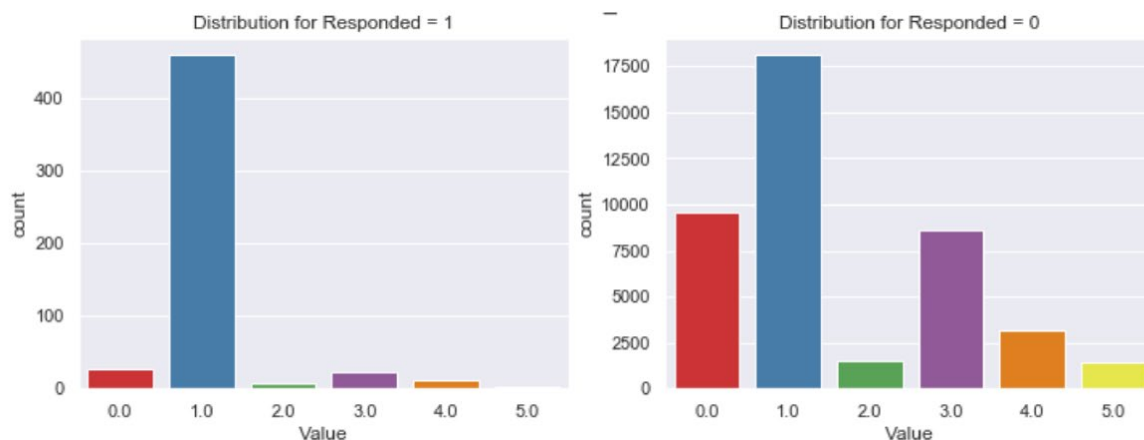
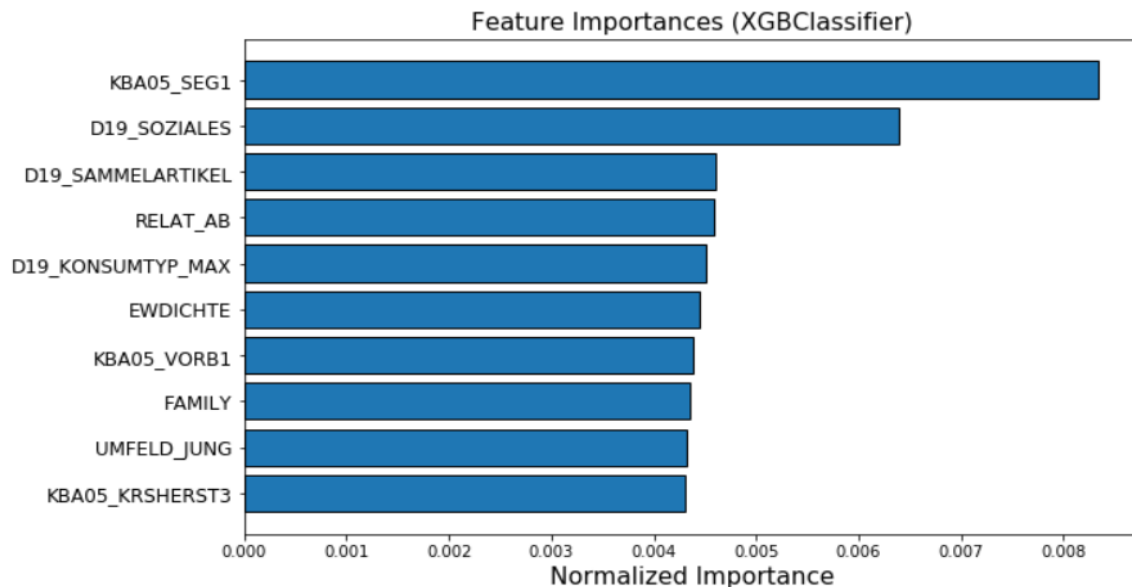
My model XGBoost has landed me on the top position among 164 global participants on Kaggle Leader Board with 81% of my model prediction being right. Hence I am choosing XGBoost as my model of choice for this project.

Below is the link for the competition leaderboard.

<https://www.kaggle.com/c/udacity-arvato-identify-customers/leaderboard>

## Justification

KBA05\_SEG1 is identified as the feature with the greatest impact in the model. Feature is related to share of very small cars (Ford Ka etc.) in the microcell



The ROC-AUC score originally obtained with LR was around 0.66 but with the optimized xgboost it went up to 0.81

I see a scope for improvement where time taken to do further testing and optimization is a hindrance now but I will be working on improving my scores which will be an ongoing part of my journey.

**Thank You!**