
Project Report

Machine Learning Engineer Nanodegree

Julio Guijarro Hernández

11-05-2020

Índice

| | |
|---|----|
| 1. Project Overview | 1 |
| 2. Problem Statement | 2 |
| 3. Metrics | 2 |
| 4. Data Exploration, Exploratory Visualization and data preprocessing | 3 |
| 5. Algorithms and Techniques | 7 |
| 6. Benchmark | 11 |
| 7. Methodology | 12 |
| 8. Results | 13 |
| 9. Justification | 14 |

Índice de figuras

| | | |
|-----|--|----|
| 1. | Data distribution in the named columns. | 3 |
| 2. | Data sample at the first read previous corrections | 3 |
| 3. | Read files function correctly formed. | 4 |
| 4. | Germany population sample | 5 |
| 5. | Arvato's customers sample | 5 |
| 6. | Nullity percentage in each column inside customers dataset . . . | 6 |
| 7. | Variable importance in the baseline model | 7 |
| 8. | Pipeline defined to perform PCA. | 8 |
| 9. | PCA components contributions in the variance. | 8 |
| 10. | Elbow method results. | 9 |
| 11. | Pipeline with the Kmeans model added. | 9 |
| 12. | Clusters of the Kmeans model with the PCA features. | 10 |
| 13. | Customers ratio inside each cluster. | 10 |
| 14. | LGBM with best score. | 14 |
| 15. | Kaggle competition ranking. | 14 |

1. Project Overview

The project presented in this document is the one chosen from those proposed by the Udacity platform to carry out the Capstone Project of the Nanodegree of Machine Learning Engineer.

The project presented in this document is the one chosen from those proposed by the Udacity platform to carry out the Capstone Project of the Nanodegree of Machine Learning Engineer.

The project proposes to carry out the detection of possible clients for its sales campaign via mail. To this end, two main steps are developed in this project: a population segmentation by means of unsupervised learning algorithms and another step consisting of the development of a supervised prediction model.

Prior to these stages, a pre-processing of the data must be performed to facilitate its management during the development of the project.

The information to be used in this project is that provided by Udacity and Arvato for the project. 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. Two more metadata files are provided:

- DIAS Information Levels – Attributes 2017.xlsx: top level list of attributes and descriptions.

- DIAS Attributes – Values 2017.xlsx: detailed mapping of data.

2. Problem Statement

The problem in this project, is about population segmentation and customer campaign prediction.

With the segmentation of the population, Arvato will aim to classify in an optimal way the population of which it has information in order to be able to consider it as a target population or not for its sales and marketing campaigns, in this specific case, for its mail order campaign.

This problem can be solved in several steps to obtain the final result of knowing who should be included in the company's sales campaign. The first of these steps would be to obtain a summary of the characteristics that define each person of whom we have information, and of this available information which is really useful for the project and its objective.

With the Kmeans algorithm, it is intended to develop a clustering of customers and that automatically seeks to detect hidden patterns at first sight within the information of the population and thus be able to analyze the aggregation component of the population.

To this end, a series of unsupervised learning techniques will be used. Specifically, first a PCA algorithm will be applied to extract the main characteristics that the variables provide to facilitate the creation of the unsupervised K-means learning algorithm. Once we have all the refined data, a supervised prediction model will be made. In this part it will be important to test several learning algorithms, such as XGBoost, LGBM or CatBoost in addition to the information with which these models are trained. It will be tested with data distributions composed by: only information from mail campaigns, only information from the population and with mixed information.

3. Metrics

In this Project a classification is going to be elaborated, so in classification models one of the most used and recommended metrics is the AUC, which will be the one I will use in this project.

Also due to the low ratio of positive cases in the data, I will check the Precision Recall AUC metric that is recommended for very small positive ratio classification cases.

4. Data Exploration, Exploratory Visualization and data preprocessing

The first step in data exploration is to read the files provided by Arvato for the project.

At first reading csv files we see the first problem with certain columns, namely the columns 'CAMEO_DEUG_2015', 'CAMEO_INTL_2015'. The problem with these columns is that they have in their data a mixture of data type, because in certain columns that are supposed to have no information of this field, we have the values 'X' and 'XX' respectively in each of the columns.

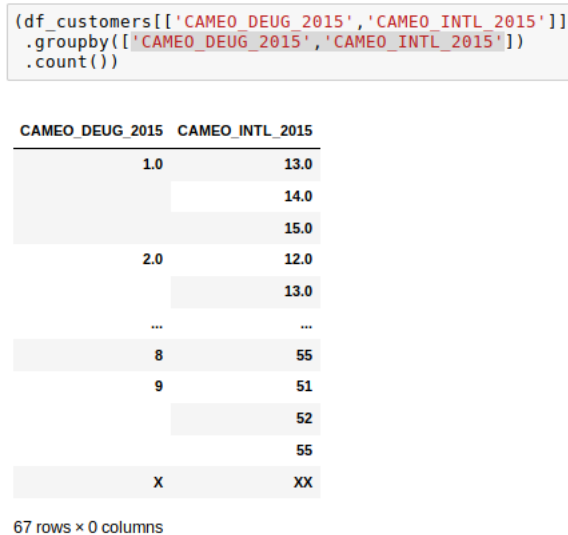


Figura 1: Data distribution in the named columns.

To solve this problem, we configured the reading of the files to identify 'X' and 'XX' as NaN values in the reading. In addition to this configuration, the first column, called 'Unnamed:0' and referring to the first index column of the file, will be deleted. Another value to be treated is -1, which in the attribute file is also associated to null values in all the attributes.

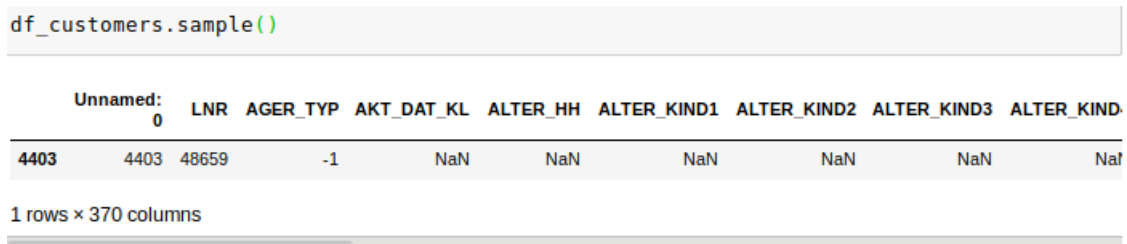


Figura 2: Data sample at the first read previous corrections

Another important aspect to deal with is the type of data that is inside the data. Thanks to the file of attributes that is provided by Arvato, we can identify the variables of categorical type that we have, which is very useful in the future for the modeling of the different algorithms to be implemented. To do this, a dictionary is generated with the variables that are in this file and they are called as 'category', except those that are indicated to be numerical, which are called of type 'float'.

Finally we read correctly the files, with this configuration:

```
df_ger_popu = pd.read_csv(PATH_DATA_GERMANY /  
                           'Udacity_AZDIAS_052018.csv',  
                           dtype= features_dict,  
                           na_values=[-1, 'X', 'XX'],  
                           index_col=0,  
                           sep=';')  
  
df_customers = pd.read_csv(PATH_DATA_CUSTOMER /  
                            'Udacity_CUSTOMERS_052018.csv',  
                            dtype= features_dict,  
                            na_values=[-1, 'X', 'XX'],  
                            index_col=0,  
                            sep=';')
```

Figura 3: Read files function correctly formed.

After this, the values of the category variables marked as 0 must be converted to a missing value, due to the information we have in the attributes file.

After these steps we can access the data read correctly.

```
: df_ger_popu.sample(10)
```

| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTEF |
|--|--------|----------|------------|----------|-------------|-------------|-------------|-------------|-------|
| | 138156 | 928337 | NaN | 1.0 | 15.0 | NaN | NaN | NaN | NaN |
| | 227815 | 350100 | NaN | 6.0 | 15.0 | NaN | NaN | NaN | NaN |
| | 23350 | 622181 | 2 | 9.0 | 6.0 | NaN | NaN | NaN | NaN |
| | 367340 | 769000 | NaN | 9.0 | NaN | NaN | NaN | NaN | NaN |
| | 386871 | 913571 | NaN | 1.0 | NaN | NaN | NaN | NaN | NaN |
| | 886579 | 286739 | NaN | 9.0 | 14.0 | NaN | NaN | NaN | NaN |
| | 274267 | 265019 | 1 | 1.0 | NaN | NaN | NaN | NaN | NaN |
| | 22006 | 769521 | NaN | 2.0 | NaN | NaN | NaN | NaN | NaN |
| | 457318 | 522893 | 2 | 9.0 | 20.0 | NaN | NaN | NaN | NaN |
| | 110605 | 943032 | 3 | 1.0 | 15.0 | NaN | NaN | NaN | NaN |

10 rows × 366 columns

Figura 4: Germany population sample

```
df_customers.sample(10)
```

| | LNR | AGER_TYP | AKT_DAT_KL | ALTER_HH | ALTER_KIND1 | ALTER_KIND2 | ALTER_KIND3 | ALTER_KIND4 | ALTEF |
|--|--------|----------|------------|----------|-------------|-------------|-------------|-------------|-------|
| | 52983 | 81861 | 2 | 1.0 | 11.0 | NaN | NaN | NaN | NaN |
| | 34681 | 139793 | 3 | 1.0 | 13.0 | NaN | NaN | NaN | NaN |
| | 149795 | 153776 | 2 | 1.0 | 19.0 | NaN | NaN | NaN | NaN |
| | 60685 | 84228 | 2 | 1.0 | 0.0 | NaN | NaN | NaN | NaN |
| | 157840 | 188265 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| | 26822 | 62676 | 3 | 1.0 | 13.0 | NaN | NaN | NaN | NaN |
| | 24757 | 130306 | NaN | 1.0 | 0.0 | NaN | NaN | NaN | NaN |
| | 45577 | 188989 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| | 185779 | 169179 | 1 | 1.0 | 8.0 | NaN | NaN | NaN | NaN |
| | 24330 | 91705 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

10 rows × 369 columns

Figura 5: Arvato's customers sample

The measurements of these datasets are:

1. Registers: 891.221
2. Columns: 366

Arvato customers:

1. Registers: 191.652

2. Columns: 369

Upon first glance, it can be seen that the customers' dataset has three more variables that define their records. These three variables are 'CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP', which are going to be eliminated from the customers dataset to align all the data later on.

4.1. Removal of columns with high nullity in the datasets.

You can see at first glance the large number of null values within the different variables that define the records. Therefore, we are going to proceed to eliminate variables that have a large amount of null values.

A first graphic analysis of the distribution of null values in the variables is performed by representing the histogram of the percentage of null values within the columns, as shown below:

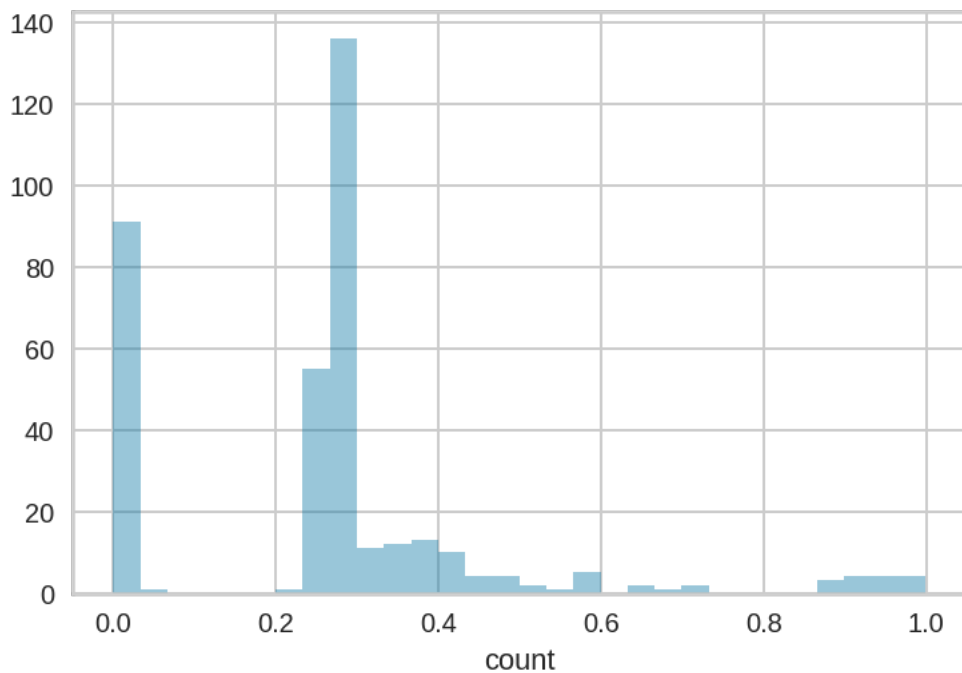


Figura 6: Nullity percentage in each column inside customers dataset

It can be observed that there are many variables with a percentage of null values less than 40%, so we will take this value as a threshold to discard variables. With this percentage, we get a total of 46 variables to be discarded.

On the other hand, a baseline model will be generated to check independently from our perception how the model learns from the data and which variables it gives a greater weight to in order to

obtain the prediction. For this purpose, I have generated a synthetic dataset with the data provided by Arvato's population and customers and I supply it to an XGBoost model modelled with the H2O package.

The importance of variables in the model are the following:

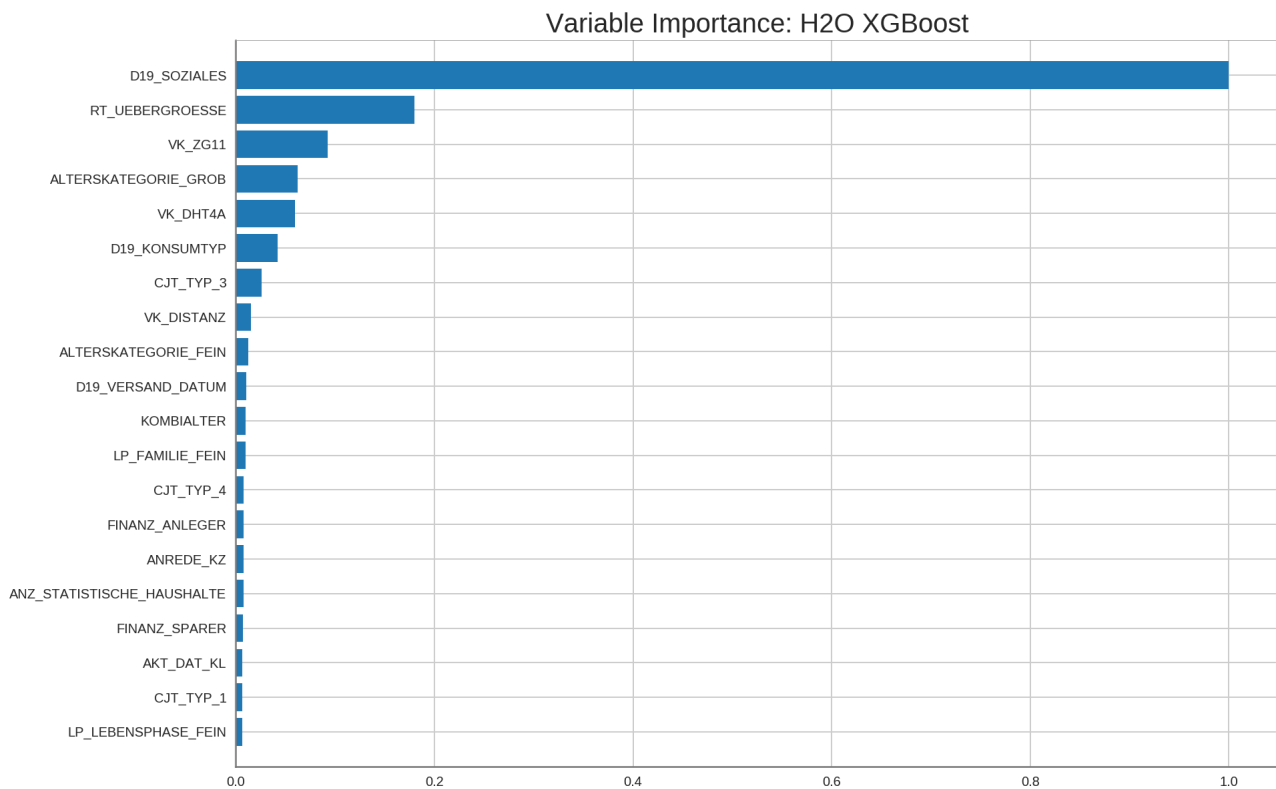


Figura 7: Variable importance in the baseline model

By comparing these variables with those collected as variables with a high percentage of null, there is no coincidence, so we can rule them out without any problem.

5. Algorithms and Techniques

5.1. PCA Model

The PCA model allows us to carry out a reduction of the dimensions and thus facilitate in future steps the clustering of the data. It is important to remember that only numerical variables are used in the PCA model.

For this we will do two more steps of data processing before feeding the model. These two steps are:

1. Fill in the NaN values that exist within the numerical variables with numerical values 'Dummies'. In this project the value of -99 will be used.
2. Normalize the variables to be able to perform the normalization in a more efficient and easy way.

To do this, a pipeline is created that collects all the steps exposed so far:

```
pipeline_pca = make_pipeline(PandasTypeSelector(include='number'),  
                             SimpleImputer(strategy='constant', fill_value=-99),  
                             StandardScaler(),  
                             PCA(n_components=40, random_state=1993))
```

Figura 8: Pipeline defined to perform PCA.

As can be seen in the image above, a 40 components PCA has been chosen to explain the variance of the data.

In the image below, the three main components are the ones that explain the most variance in the data, and then the next ones decrease in variance, gradually obtaining the 90 % variance we are looking for.

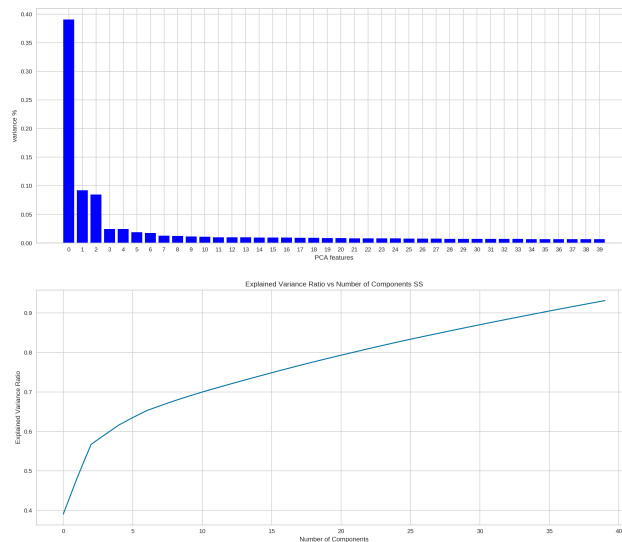


Figura 9: PCA components contributions in the variance.

Once the model has been adjusted to all the data we work with, I divide it into training and testing data to carry out the clustering of the data and check that the results are correct.

5.2. Kmeans model

To elaborate the Kmeans model, first I perform the elbow method, to get to choose the number of clusters to implement:

```
Kmeans_model = KMeans()
elbowMethod = KElbowVisualizer(Kmeans_model, k=(1,7))

elbowMethod.fit(X_train_transformed)
elbowMethod.show()
```

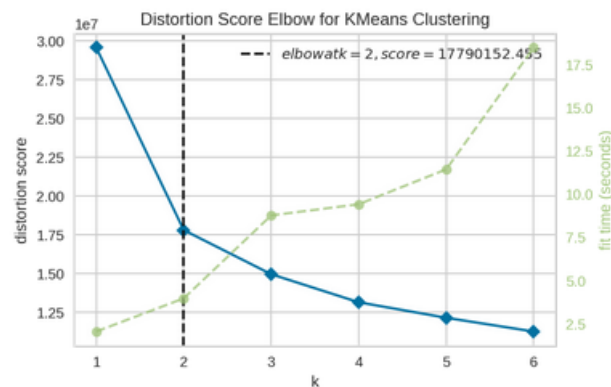


Figure 10: Elbow method results.

According to this, the optimal number of clusters to perform in the Kmeans model is 2. Therefore, we implement this model within the pipeline and train it with a part of the total data (train data), testing it later with the remaining data (test data).

```
pipeline_kmeans = make_pipeline(pipeline_pca,
                                KMeans(n_clusters=2, random_state=1993))

pipeline_kmeans.fit(X_train)
```

Figure 11: Pipeline with the Kmeans model added.

In the next figure, we can see a good segregation in cluster of population.

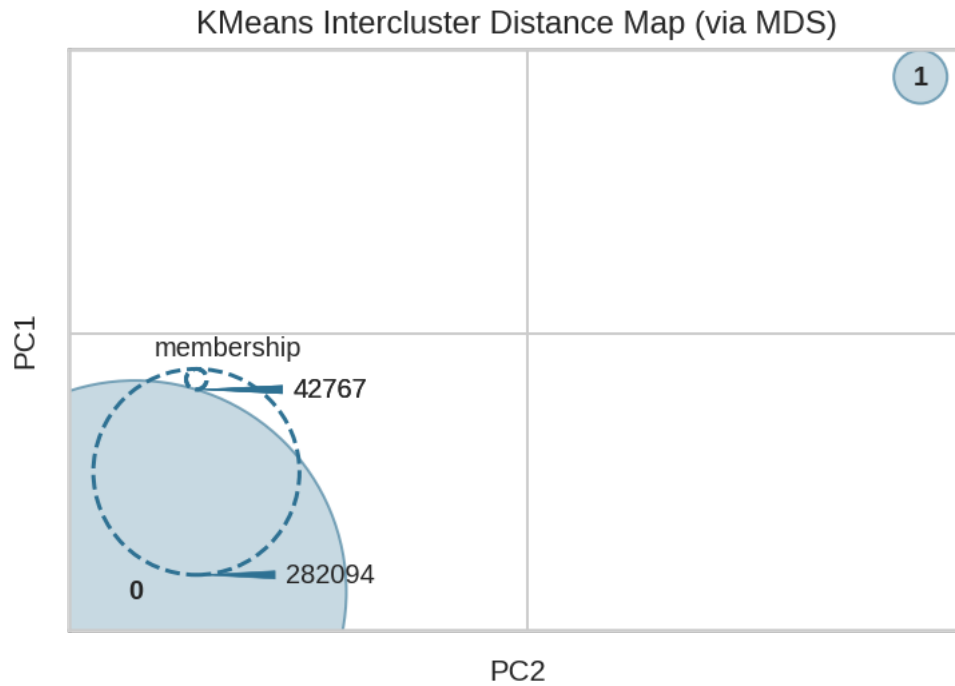


Figura 12: Clusters of the Kmeans model with the PCA features.

Now I look at the information that the clusters have, I check the ratio of people who are customers against those who are not in each of them. Both for training and for testing, we see a larger rate within the cluster 1, the smallest than in the 0.

Mean of the target customer in each
cluster and volume in train data

| | cluster | target_mean | volume |
|---|---------|-------------|--------|
| 0 | 0 | 0.151191 | 282094 |
| 1 | 1 | 0.347113 | 42767 |

Mean of the target customer in each
cluster and volume in test data

| | cluster | target_mean | volume |
|---|---------|-------------|--------|
| 0 | 0 | 0.150638 | 657704 |
| 1 | 1 | 0.349743 | 100308 |

Figura 13: Customers ratio inside each cluster.

So the population segmentation has given us optimal results, which will add more information about the records.

5.3. Supervised learning models

In addition to the models I have presented so far, a number of models have also been used for the customer detection part of the company's mail campaign, especially those known to have a better performance when working with tabulated data, such as:

1. XGBoost.
2. Catboost.
3. LGBM.

These models are going to be used for the elaboration of the predictions on the mail campaigns that the company carries out. In the models, not only the hyperparameters that compose it are studied, but also the type of data that feeds it.

In this project it has been used:

1. Training only with population data, taken as the baseline of the project.
2. Training with previous mail campaigns of the company of which information is already available.
3. With mixed data from the past mail campaign, of which we know when it had success in which customers, as in the total information of the population, which differentiates between Arvato's users customers and general population of Germany.

In the project, it was concluded that the best result was obtained with mixed information and by using a LGBM. This information will be presented in the results section.

6. Benchmark

To determine the improvement over the hyper-parameterization of the models and the data used, I have chosen to use the result of the first model to obtain the variables of the model at first, which uses population data simply to predict on the test data than in the Kaggle competition of the project. The AUC obtained is 0.724.

7. Methodology

7.1. Data Preprocessing

The data preprocessing throughout the project has already been explained throughout this report. As a recap:

- When reading data, a variable cleaning has been performed with mixed data columns with values such as 'X' and 'XX', besides converting values from -1 to NaN. In the categorical variables this process is also done for values of 0.
- When reading and cleaning the data, the variables that are read from the attributes file are also taken into account to identify the categorical variables in the data and to identify them as 'category'.
- In the unsupervised learning part, in the pipeline both the selection of only numerical variables, which are those used for the PCA algorithm, and the filling in with a numerical value of the null values are performed.
- The Supervised Algorithms, in the part where the models are trained only with data from old mail campaigns, an Oversampling of the positive cases is performed to achieve a better ratio when training. In addition to this, an encoding of the categorical variables is carried out. For this project, the best performance has been shown to be the WOE (Weight Of Evicende) coding.

7.2. Implementation and Refinement

For the development of all the algorithms and processes that have been exposed until now, the python language has been used and two notebooks have been developed for their development and monitoring. The notebooks are:

1. 1_Preprocessing_and_unsupervised_algorithms
2. 2_ML_algorithms_for_supervised_learning

In addition, the file `helpers_data_cleaning.py` has been used for cleaning and refining the data

Those parts related to unsupervised learning have already been explained above and all the points discussed.

On the part of the supervised algorithms, their implementation and refinement have been the ones that have required more tests throughout the development of the project. In this part, 3 models are implemented: XGBoost Catboost and LGBM. Each of them have been trained with different datasets:

7.2.1. Mailout campaign data

At first, the models are entered with the data provided by Arvato. These first models with the untreated data do not have a very good result. In order to improve it, an oversampling of the data is carried out to improve the rate of positive cases that the model can capture. With this change, the performance of the model improves.

7.2.2. Mailout campaign and population data mixed

To improve the performance of the model and increase the positive cases of clients that the model observes, the entire email campaign is mixed with random records taken from the total population.

To avoid the population information taking too much weight on the records of the mail campaign, a percentage is used. After several tests, the optimal percentage of records to be entered seems to be around 15 % of population records. With this modification in the data a better classification within the model is achieved.

8. Results

The final model used for the Kaggle competition associated with this nanodegree was the LGBM model with a parameterization:


```

: param_lgb = {
    "ntrees" : 50,
    "max_depth" : 8,
    "learn_rate" : 0.01,
    "sample_rate" : 0.8,
    "col_sample_rate_per_tree" : 0.8,
    "min_rows" : 3,
    "random_state": 1993,
    "stopping_metric": 'AUC',
    "n_jobs":-1
}

pipeline_encoder_and_lgb_mixed = make_pipeline(
    WOEEncoder(cols=cat_feats),
    lgb.LGBMClassifier(**param_lgb)
)

```

Figura 14: LGBM with best score.

and using the mixed data from mail campaigns and the general population at 15 %, an AUC of 0.80954 was obtained in this project.

9. Justification

Thanks to the model described above, it has been possible to obtain the second position of the Kaggle competition that this Nanodegree raises:



| # | Team Name | Notebook | Team Members | Score ? | Entries | Last |
|--|--------------------------|----------|---|---------|---------|------|
| 1 | Ambresh Patil | |  | 0.81063 | 58 | 3mo |
| 2 | Julio Guijarro Hernandez | |  | 0.80954 | 16 | 4d |
| Your Best Entry ↑ Your submission scored 0.80073, which is not an improvement of your best score. Keep trying! | | | | | | |

Figura 15: Kaggle competition ranking.

So I think this project has a very good performance and has satisfied the expectations that were at first.

As possible future work to further improve the model could investigate further hyperparameters of the models as well as generate synthetic variables from the variables already present and see if the performance improves.