Machine Learning Engineer Nanodegree

Capstone Project

Nathália Brunet
(nathaliabrunet@gmail.com)

September 14th, 2020

Customer Segmentation Report for Arvato Financial Solutions

Project Overview

Arvato is a business services company operating in different parts of the world, offering financial solutions in the most diverse forms, from payment processing to risk management activities, with development always focused on innovation. Arvato is owned by Bertelsmann, which is a media, services and education company operating in more than 50 countries and one of its priorities is to value solutions with an emphasis on creativity and entrepreneurship [1].

In this context, it is interesting to understand the customer segments that the company serves. For a company with a large customer base, the customer segmentation stage is very important. It aims to obtain more information (and really relevant information) from its customers, which can be used for a more personalized and targeted service, for example.

Problem Statement:

The goal is to perform a segmentation of customers, and with the knowledge obtained, identify whether a customer will respond to a specific marketing campaign or not. Thus, it is possible to increase the effectiveness of the campaign by directing it to an audience with a greater potential to " be reached" the campaign. In summary, the goal is to improve the performance of marketing campaigns.

Datasets and inputs:

The datasets were provided by Bertelsmann/Arvato in the context of the Udacity Machine Learning Engineer Nanodegree. It is a data that represents a real-life data science task.

The problem can also be found in the Kaggle challenge [2].

There are four data files associated with this project:

1. Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns).
2. Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns).
3. Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).
4. Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).

For more information about the columns depicted in the files, it is possible refer to two Excel spreadsheets the are provided too:

1. DIAS information levels - attributes2017.xlsx: is a top-level list of attributes and descriptions, organized by informational category
2. DIAS Attributes - Values 2017.xlsx: is a detailed mapping of data values for each feature in alphabetical order

---

Solution Statement:

Initially, data pre-processing is performed. With the data processed, I continued with the analysis, which is divided into two parts.

1. The first part is done to identify if the costumer are a potential customer or not. Through unsupervised learning, to segment customers into significant groups, identifying clusters / segments of the population in general that best correspond to the base of interest.

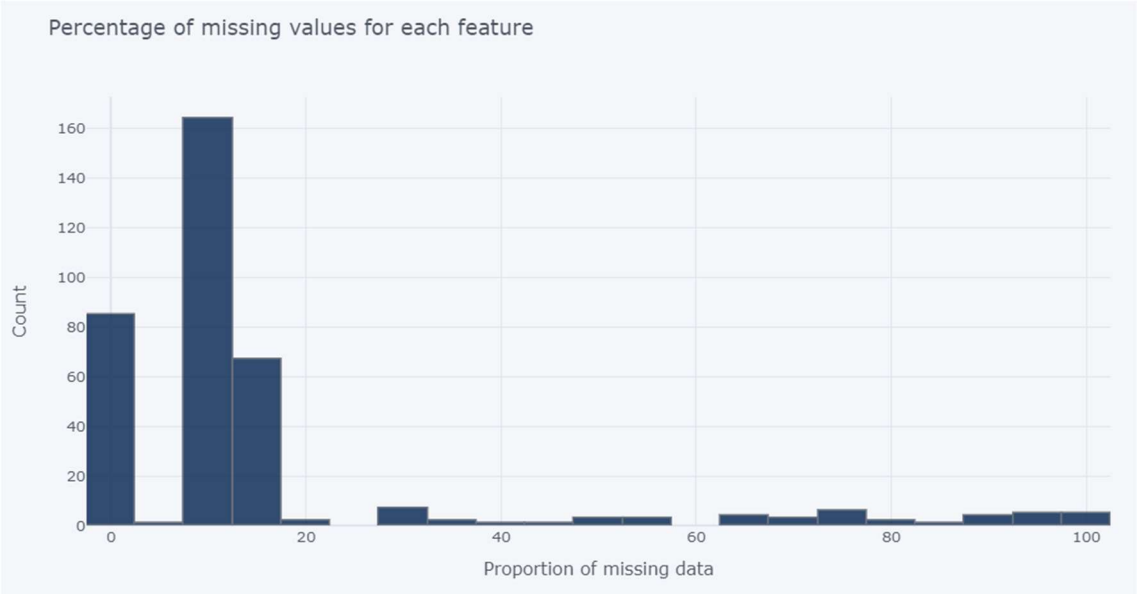
2. With the first part completed, and the segments of the customers identified, supervised learning will be approached, with the objective of forecasting the individuals most likely to become customers. In this stage could be test some techniques that are good supervised learning, which may include Logistic Regression, XGBoost and LightGBM models, for example.

Data pre-processing

For exploratory analysis, and data cleaning, some points were followed:

- Encode missing values as NaNs
- Remove columns that have 20% of missing values

To define the value = 20%, I plotted the proportion of missing values by features and analyzed the behavior of the graph. It is possible to observe that most of the base has less than 20% of the missing data, so I chose this value as the threshold point. Figure with percentage of missing values for each feature:



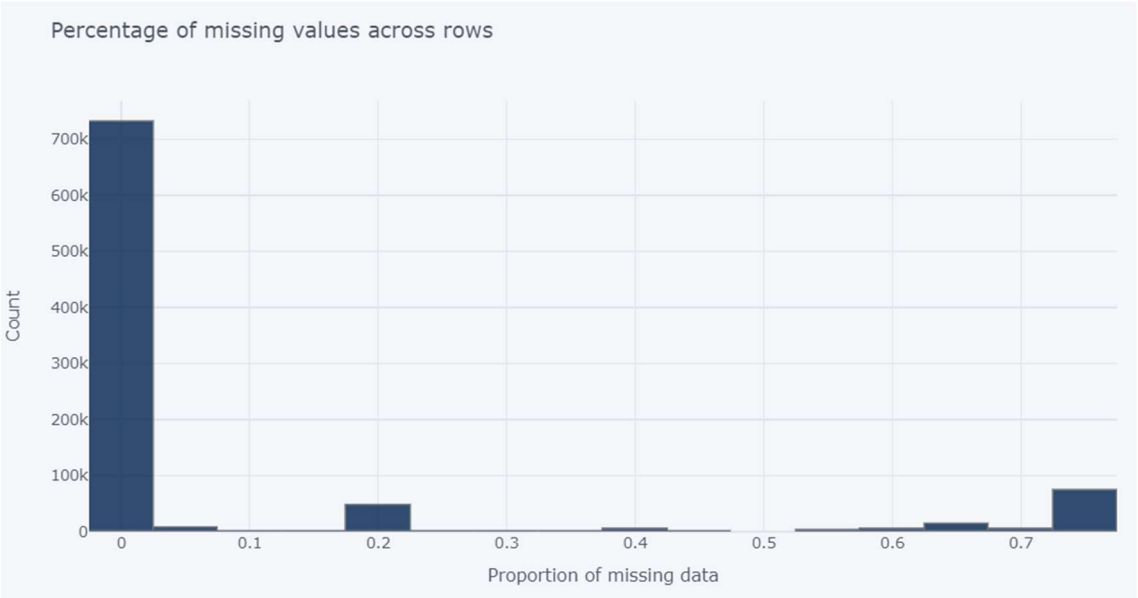
Note

- Before: 366 columns
- After drop: 319 columns

47 columns (13%) were dropped

- Remove rows that have more than 5% of missing values per row.

In relation to rows, the analysis is similar. To define the value = 5%, I plotted the proportion of missing values per row and analyzed the behavior of the graph. It is possible observe that the most part are below 5% (~ 16 missing values per row). So, i used this threshold as a criterion. Figure with percentage of missing values across rows:



Note

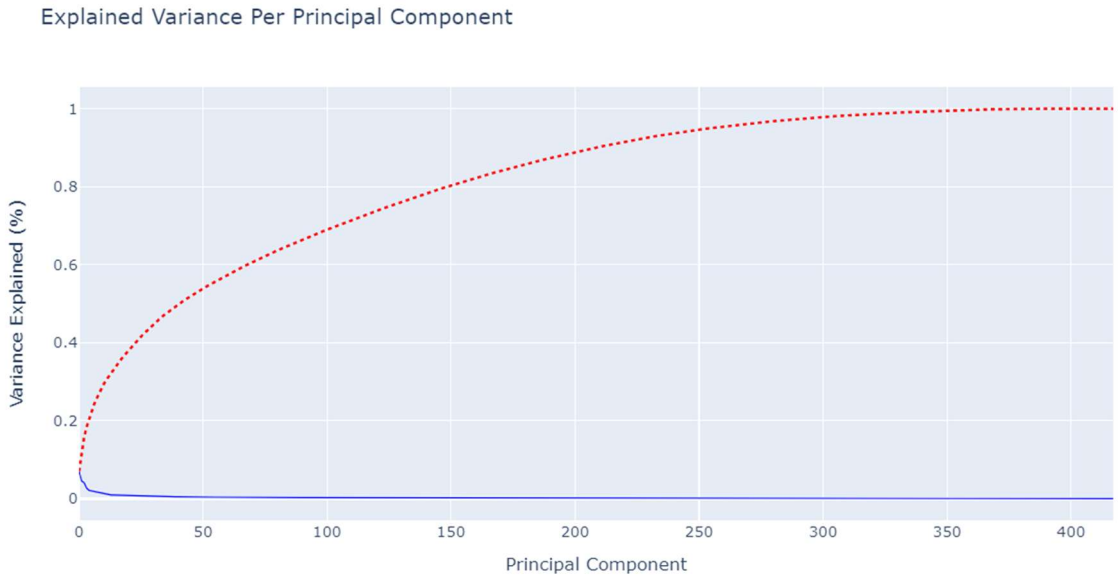
- Before: 891221 rows
- After drop: 732895 rows

158.326 rows (17.7%) were dropped

- Transformation of data types
- Encode categorical variables - Create dummies
- Generate a function to replicate what was done for the customers dataset
- Impute Missing Values
- Feature scaling, using StandardScaler

Part 1. Customer Segmentation using Unsupervised Machine Learning

After the cleaning part, the resulting dataset contains 418 columns. It is still a base with high dimensioning. In this point, was implemented PCA (Principal Component analysis) to reduce dimension to unsupervised learning, searching for vectors of maximal variance in the data. In the Figure below we have the graph explained variance per principal component. Analyzing the graph, it is possible to see that after 200 we have little variation in the value explained.

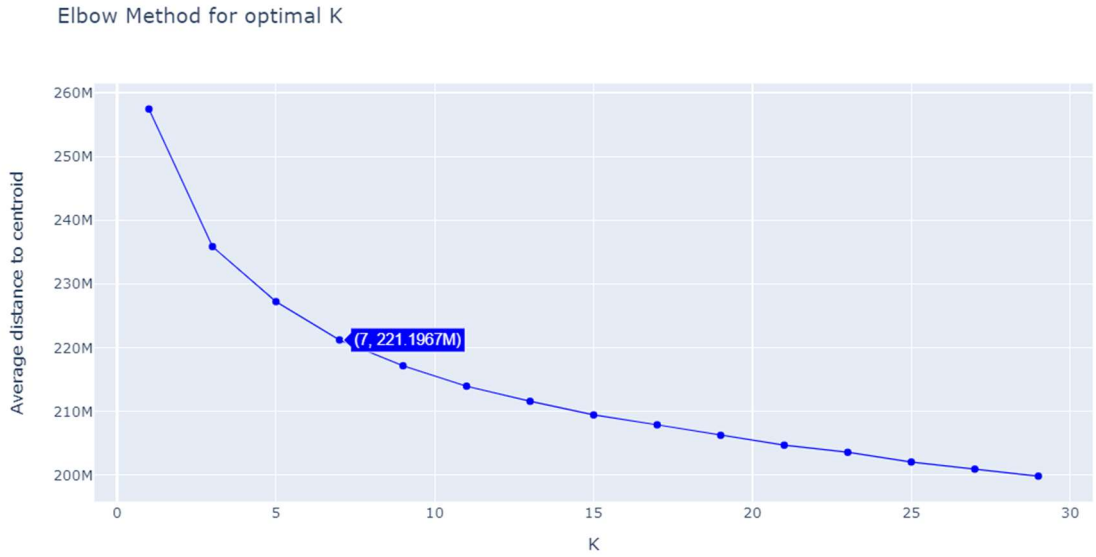


To facilitate the analysis, below are explained variance values between of 80% and 86%.

```
Looks at Explained variance between 80% and 90%:
Number of components: 149 --- Explained variance:  80.0 %
Number of components: 150 --- Explained variance:  80.0 %
Number of components: 151 --- Explained variance:  81.0 %
Number of components: 152 --- Explained variance:  81.0 %
Number of components: 153 --- Explained variance:  81.0 %
Number of components: 154 --- Explained variance:  81.0 %
Number of components: 155 --- Explained variance:  81.0 %
Number of components: 156 --- Explained variance:  81.0 %
Number of components: 157 --- Explained variance:  82.0 %
Number of components: 158 --- Explained variance:  82.0 %
Number of components: 159 --- Explained variance:  82.0 %
Number of components: 160 --- Explained variance:  82.0 %
Number of components: 161 --- Explained variance:  82.0 %
Number of components: 162 --- Explained variance:  83.0 %
Number of components: 163 --- Explained variance:  83.0 %
Number of components: 164 --- Explained variance:  83.0 %
Number of components: 165 --- Explained variance:  83.0 %
Number of components: 166 --- Explained variance:  83.0 %
Number of components: 167 --- Explained variance:  84.0 %
Number of components: 168 --- Explained variance:  84.0 %
Number of components: 169 --- Explained variance:  84.0 %
Number of components: 170 --- Explained variance:  84.0 %
Number of components: 171 --- Explained variance:  84.0 %
Number of components: 173 --- Explained variance:  85.0 %
Number of components: 175 --- Explained variance:  85.0 %
Number of components: 176 --- Explained variance:  85.0 %
Number of components: 177 --- Explained variance:  85.0 %
Number of components: 178 --- Explained variance:  85.0 %
Number of components: 179 --- Explained variance:  86.0 %
Number of components: 180 --- Explained variance:  86.0 %
```

So, using 173 components we can explain 85% (highlighted in blue box) of the variability in the original data.

Then, I performed the clustering using *k-means*. I used the Elbow method to choose the number of k (clusters). A bad value for k: would be one so high that only one or two very close datapoints are near it, or one so small, that datapoints are far away from the center. The elbow method takes a value for k and looks at the average distance of each of your data points to the cluster.

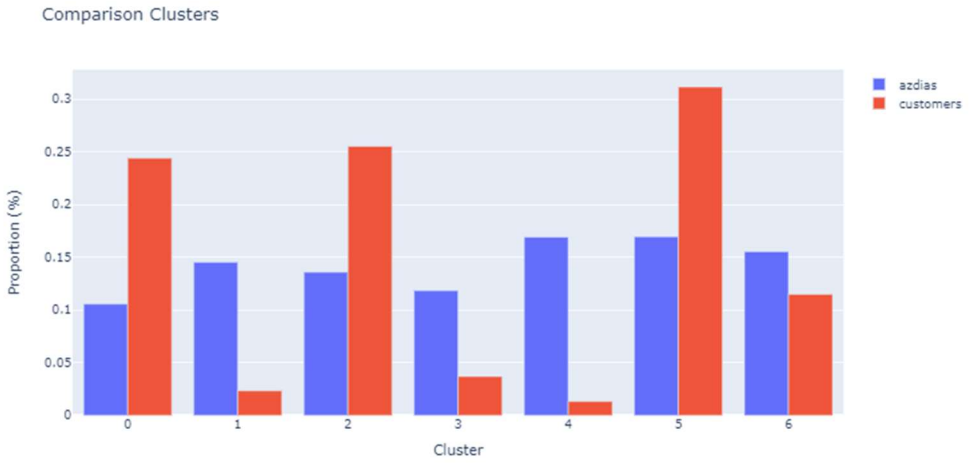


After applying the algorithm with **K** ranging from 1 to 30, with step = 2 (k=1, k=3, k=5, k=7…). So, with the analysis I decided to continue with 7 clusters.

Running k-means with 7 clusters for the two data sets (AZDIAS and CUSTOMERS), it was possible to make a comparison, for this, I calculated the quantities (and their proportions) for each cluster, as can be seen in the figure below:

```
Counts per clusters for AZDIAS dataset (on the graph the values are in percentages)
 5    124354
4    124046
6    113952
1    106602
2     99665
3     86791
0     77485
Name: Clusters, dtype: int64

Counts per clusters for CUSTOMERS dataset (on the graph the values are in percentages)
 5    40906
2    33519
0    32052
6    15088
3     4847
1     3069
4     1739
Name: Clusters, dtype: int64
```



It is possible observe that the cluster 5 is overrepresented in the customer data (~31%), and the cluster 4 is underrepresented (~1.3%).

Part 2. Supervised Learning Model

At this moment we want use supervised training techniques to identify an individual's chance to respond to a marketing campaign positively.

In the train dataset we have 42,982 rows with 367 columns (features), remembering that have the "RESPONSE" columns, which includes the answer whether the individual became a customer of the company or not. And we have the test dataset, with 42,833 people.

I implemented 3 different machine learning models to compare the results, Logistic Regression, XGBoost and LightGBM.

- Logistic Regression: it is a statistical method, in which the probability of the occurrence of an event is calculated to classify two classes, being one of the most used machine learning algorithms for this type of problem. It can be used for several types of classification problems, such as spam detection, disease prediction (such as diabetes and cancer), and one that I believe is very similar to

the objective of our problem, quality classification problem and interest of visitors who browse the sites [3]. For example, a company wants to know if the visitor to the site has a genuine interest in the products offered or if they have entered the site for other reasons, with the classification of these visitors, you can do differentiated retargeting actions, present campaigns and personalized actions according to the profile or direct investments in an intelligent way, similar to our case, in which we want to classify whether it is a potential client or not.

- XGBoost: is a machine learning algorithm, based on a decision tree and that use a Gradient boosting structure. Is one of the most popular machine learning algorithm these days, and has been the most successful in Kaggle (an online community of data scientists and machine learning practitioners that offer machine learning competitions). In addition, it has also been used, with sucessfully, in differents applications in the industry [4].
- LightGBM: is a fast gradient boosting framework, based on a decision tree algorithm. It is used for several types of machine learning problems, including classification problems. In [5], experiments showed that LightGBM can accelerates the training process of the conventional Gradient boosting decision tree by more than 20 times, and reaching almost the same accuracy.

We can observe in Figure that 98.76% (42.430) of the data refer to individuals who did not respond to the campaign, with only 1.24% (532) of the individuals who responded to the campaign, which means that the data training are very imbalanced, so accuracy will not be an evaluation metric, because can gives inaccurate and misleading information about the classifier.

```
print('Count per class:\n', mailout_train['RESPONSE'].value_counts(normalize = False))

color='#00224e'
mailout_train['RESPONSE'].value_counts(normalize = True).iplot(kind='bar',
                                                               color= color,
                                                               title = 'Dataset imbalanced',
                                                               xTitle = 'RESPONSE',
                                                               yTitle = 'Percentual')
```

```
Count per class:
 0    42430
1      532
Name: RESPONSE, dtype: int64
```

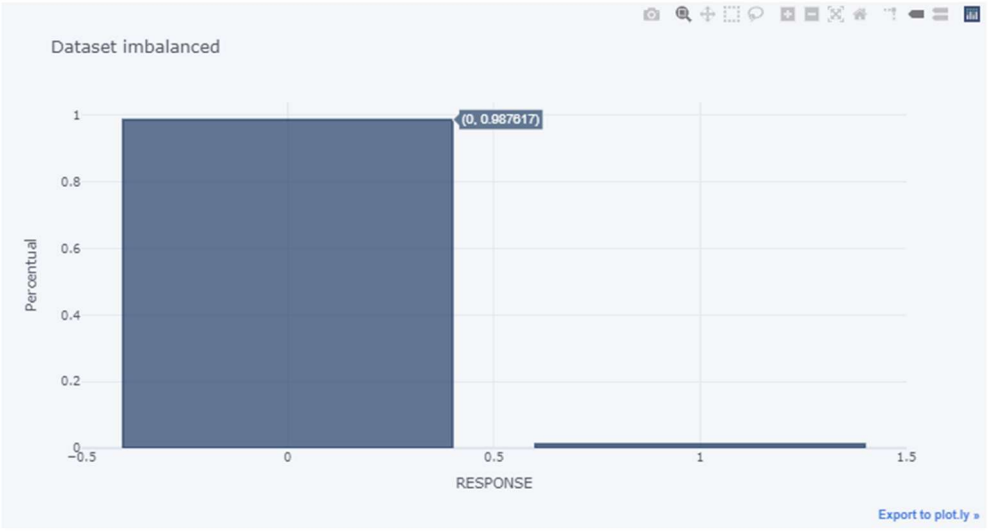


Figure 1 – Dataset Imbalanced

Many researchers adopt the Receiving Operating Characteristic (ROC) curve as a metric. The idea is to estimate the performance of a binary classifier by varying the discrimination threshold [6]. Knowing this, added that kaggle also uses this metric in competition, ROC AUC was used as a *metric* for choosing the best model.

Table with summary of results for comparison:

	Model	ROC_AUC
1	([DecisionTreeRegressor(criterion='friedman_ms...	0.693565
2	LGBMClassifier(max_depth=3, n_estimators=8)	0.692253
0	LogisticRegression(max_iter=250)	0.615380

As observed in the table, Gradient Boosting Classifier had the highest ROC AUC of 0.693, fol lowed closely by the LGBMClassifier, with an ROC AUC score of 0.692, we can accept that it was a tie, they performed almost equal. As mentioned before, that experiments showed that LightGBM can accelerates the training process of the conventional Gradient boosting decisi on, and reaching almost the same accuracy, was confirmed in that test. See the results with y our execution times:

```
Time execution: 40.293681383132935

ROC_AUC: 0.6153796893390602
Model: LogisticRegression(max_iter=250)
----

Time execution: 392.23408675193787

ROC_AUC: 0.693564718895765
Model: GradientBoostingClassifier(max_depth=2, n_estimators=9, random_state=0)
----

Time execution: 128.1362018585205

ROC_AUC: 0.6922526986787038
Model: LGBMClassifier(max_depth=3, n_estimators=8)
----
```

LightGBM has almost the same ROC AUC score, but about 3 times faster, 128 seconds compared to 392.

Conclusion

This project had the goal had the objective of accomplishing a segmentation of customers, and with the knowledge obtained, identify whether a customer will respond to a specific marketing campaign or not. Thus, it is possible to increase the effectiveness of the campaign by directing it to an audience with a greater potential to " be reached" the campaign.

The datasets were provided by Bertelsmann/Arvato in the context of the Udacity Machine Learning Engineer Nanodegree. It is a data that represents a real-life data science task. Supervised and unsupervised learning methods were used to solve the project.

Principal Component Analysis (PCA) was performed to reduce dimensionality, and together with K-means grouping to create a demographic segmentation to have the comparison of the general data and the customer data.

The result of the supervised learning was not so good, some points can be reevaluated in order to improve the model, for example:

- Reassess the choice of PCA components, perhaps increasing the number.
- Evaluate the imputer method, test 'mean', for example.
- An important point is related to the imbalance of data, search different methods to deal with this issue.
- Focusing more on understanding the features, with a better understanding of them would facilitate some choices, such as what to consider when imputing missing data.
- Test other techniques in the supervised learning step.

Links:

[1] www.bertelsmann.com

[2] https://www.kaggle.com/c/udacity-arvato-identify-customers

[3] https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python

[4] https://www.datageeks.com.br/xgboost/

[5] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 3149-3157.

[6] Ali, Aida & Shamsuddin, Siti Mariyam & Ralescu, Anca. (2015). Classification with class imbalance problem: A review. 7. 176-204.