MACHINE LEARNING ENGINEERING NABODEGREE

ARVATO CAPSTONE PROJECT

1 DEFINITION

1.1 Project Overview

Historically, the question about how to keep old customers and convert more people to new customers has always been one of the most important questions for the large-scale retail business company. They frequently used sales, promotion, or marketing campaign to attract more people and increase revenue. Traditionally, senior managers used their worked experience and reports to make the decision about how to implement the promotion or marketing campaign. This method essentially relies on the human brain to select and analyze data. Machine learning algorithms are more powerful. The computer selects the most important factors by analyzing historical data, and fit variables to find the best way to predict the effects of promotions. The machine can easily handle hundreds, thousands of features to improve the prediction accuracy which cannot be done by the human brain. The introduction of Machine Learning to the sales area, can help the retail company increase revenue and reduce cost based on a more scientific plan.

The project was provided by the Arvato company. All the necessary datasets were also provided by Arvato which include four datasets as following:

Udacity_AZDIAS_052018.csv

Udacity_CUSTOMERS_052018.csv

Udacity_MAILOUT_052018_TRAIN.csv

Udacity_MAILOUT_052018_TEST.csv

1.2 Problem Statement

The client is a Germany Mail Order company. We want to have a clear vision of the composition of the current Germany population, and figure out which segmentation of the population is more likely to become our customer. Given the customer data we have, create a customer segmentation report to help understand our customers. Based on our understanding of customer data, create a model to support the email campaign.

The project focused on two major problems. First, make a customer segmentation report for the provided customer data, to help us understand the characteristic of our customer groups. Those characteristics can then be used to find potential customers in the future. The K-means algorithm will be implemented for the customer segmentation part. Second, create a machine learning model to predict the response of the email receiver and sent out emails to those with positive feedback. This will help the company acquire more customers. The SVM algorithm will be implemented for predicting the customer’s response to our email campaign.

1.3 Metrics

To evaluate the performance of our customer response predicting model, specific metrics should be defined. Given the fact that this is a skewed dataset with an imbalanced rate = 1/100, the traditional metrics like accuracy rate will cause overfit to a single class. In this situation, some metrics like AUC, F1 Score, confusion matrix can be a good way to evaluate our model. Here we use AUC as the evaluation metrics, this is also being used in the Kaggle competition.

2 Analysis

2.1 Data exploration

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).

The four datasets all have the same 366 columns, some of them have extra columns. (for example, the "CUSTOMERS" file contains three extra columns: 'CUSTOMER_GROUP', 'ONLINE_PURCHASE', and 'PRODUCT_GROUP'.) For the extra columns, we will only use the RESPONSE column from TEST.csv as y.

Dataset Characteristics:

1.The raw datasets are CSV files, can be easily read-in with python.

2.LNR column can be considered as the row index. It is a unique customer ID.

3.Except for LNR and several columns about time, all the other columns are categorical variables.

4.Most of the columns are numerical value, except for the ‘CAMEO_DEU_2015’(string) and ‘CAMEO_DEUG_2015’(object). They will all be converted to numerical value during the data cleaning process.

5. Udacity_MAILOUT_052018_TRAIN is an extremely skewed dataset which has an imbalance rate = 1/100

6. The value missing rate is high for some of the columns. Those high missing rate columns will not be used in our model. They will be filtered out during the data cleaning process.

2.2 Exploratory Visualization

Raw Dataset example:

In [4]:
```
mailout_train.head()
```

Out[4]:

	LNR	AGER_TYP	AKT_DAT_KL	ALTER_HH	ALTER_KIND1	ALTER_KIND2	ALTER_KIND3	ALTER_KIND4	ALTERSKATEGORIE_FEIN	ANZ_HAUSHALTE_AKTI
0	1763	2	1.0	8.0	NaN	NaN	NaN	NaN	8.0	15.
1	1771	1	4.0	13.0	NaN	NaN	NaN	NaN	13.0	1.
2	1776	1	1.0	9.0	NaN	NaN	NaN	NaN	7.0	0.
3	1460	2	1.0	6.0	NaN	NaN	NaN	NaN	6.0	4.
4	1783	2	1.0	9.0	NaN	NaN	NaN	NaN	9.0	53.

5 rows × 367 columns

Figure 1 – Raw Dataset

Take the Udacity_MAILOUT_052018_TRAIN.csv as an example, we can see for some columns like ALRER_KIND1, the missing rate will be high. Those columns will cause trouble for our model and will be filtered.

One of the important customer attributes is ‘CAMEO_DEU_2015’ which is string data. After I convert it into numerical data, the number stands for the CAMEO segmentation of a person’s family.

```
sns.countplot(y='CAMEO_DEU_2015',data=mailout_train)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8a4696668>
```

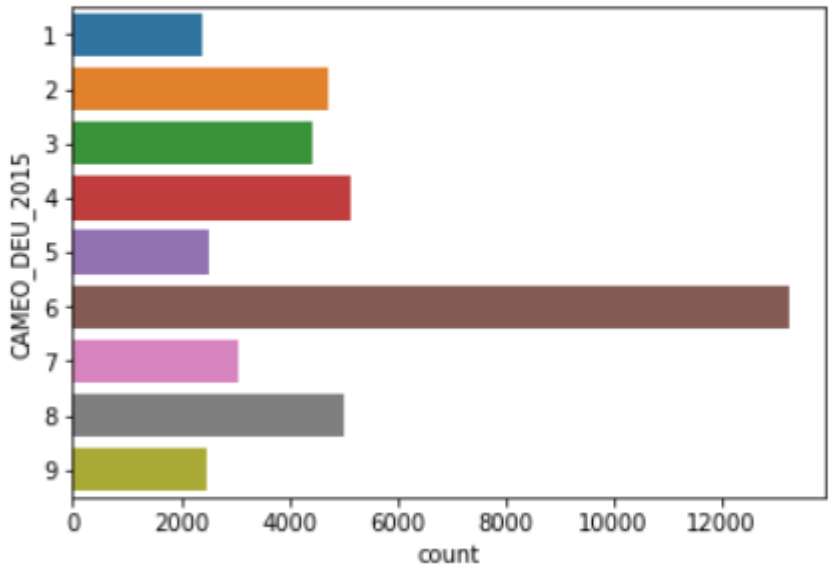

Figure 2 – CAMEO Column Count Plot

We can see a lot of people belong to section 6. In this section, the people’s family are Jobstarter, Petty Bourgeois, Long-established, Sportgardener, Urban Parents, Frugal Aging. People from this section have a small amount of spare money in their hands and will be more frugal in life. They are the biggest section of our train dataset.

2.3 Algorithms and Techniques

Data Cleaning Process: The raw dataset has 366 columns, a high missing rate for some columns, extremely skewed two classes. Some techniques were used to deal with them.

1.366 columns were filtered into 114 columns, those 114 columns are more related to our problem and relatively low correlation. PCA process will be used to help reduce the data dimension. The explained variance for the PCA process should be higher than 80%.

2. The high missing rate column should be filtered out from our analysis. The missing value will be replaced by the average value. Column missing rate higher than 25% will be ignored by our analysis.

3. SMOTE and other techniques will be used to solve the skewed dataset problem. One of the important hidden parameters for SMOTE is the k_neighbors, which is the number of nearest neighbors used to construct synthetic samples. I used the default value 5.

4. Customer Segmentation Report: To analysis the characteristic of our customers, one of the best unsupervised-learning methods is K-means. After the data engineering process, the customer data will be used to create the K-means model. The number of groups will be an important parameter to consider. I will start from two and use silhouette_score to determine the best group number.

5. Customer Response Predicting Model: This is a binary classification problem. Neural Network, SVM, Random Forest are all good choices. Here I choose SVM for my model because it should have better performance than Neural Network when dealing with a highly skewed dataset.

2.4 Benchmark

This is a Kaggle competition project, I can get a scoreboard from the following Kaggle competition page. The scoreboard has the AUC scores from more than 200 teams. I can use this scoreboard as the benchmark for measuring my model's performance. Here is the link to the competition scoreboard:

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

3 Methodology

Here is a project flow chart to show us the overview of the Arvato project.

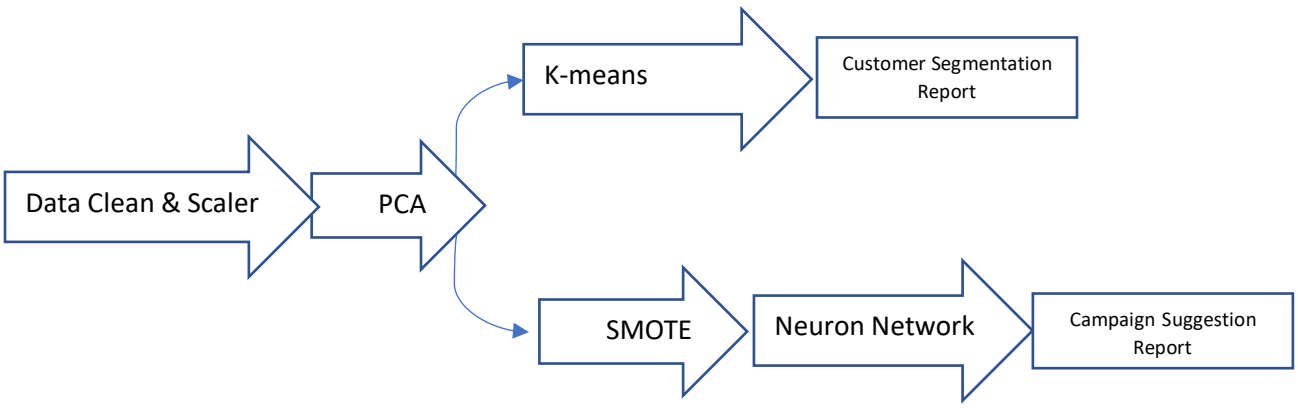

Figure 3 – Project Flow Chart

3.1 Data Processing

Here are the detailed data processing steps used in the code:

1. Read in raw data from the provided storage location from Udacity workspace. (SVM)
2. Convert string to numeric data for column 'CAMEO_DEU_2015'. This column is important for our model. A little tweak was made to consider the nine big customer segmentation. (for example: '5A', '5B', '5C' are all converted to 5) instead of the detailed segmentation.
3. Covert all missing value to -1. The raw dataset was poorly managed with some columns use -1 for missing value, but others are using 0 or 9 for the missing value. After investigating the provided metadata file (DIAS Attributes - Values 2017.xlsx), I created two lists to help me convert all missing/unknown value to -1. Also, the empty cell in the raw dataset will be filled with -1. After these processes, -1 only means missing/unknown.
4. The raw dataset has 366 columns, many of them are highly correlated or irrelevant to our problem. After carefully investigating the metadata, 114 columns were selected from the 366 columns. The col_list stored the column list information.
5. Drop out the columns with a high missing rate. The high missing rate column will not provide us valuable information, on the contrary, may downgrade our model's performance. The 'drop_missing_val_column' function was created to filtered out columns that have a missing rate exceeds the threshold. For the SVM model, the threshold was set to 25%.
6. For the remaining columns, to use average value to replace the missing value, I create a separate helper package. (included in the submission files)
7. Before using PCA, the standardize the scale of numerical columns are essential steps. I used the MinMaxScaler function from the sklearn. preprocessing package to change the data range into [0,1].
8. PCA process was used to reduce the data dimension. Function 'explained_var' was created to determine the n_componets parameter for PCA. Finally, n_componets was set to 50 and 85% variance was captured.
9. SMOTE and RandomUnderSampler techniques were introduced to deal with the imbalance effect. The sampling_strategy parameters were set to 0.0189 for the SMOTE process and the RandomUnderSampler process. (This step only used in SVM model)
10. The train/test split process randomly selects 30% data from the training dataset for testing. (This step only used in SVM model)

3.2 Implementation

3.2.1 Customer Segmentation Report

Algorithms: K-means.
K-means clustering is the most famous clustering algorithm. Because of its simplicity and efficiency, it is the most widely used among all clustering algorithms. Given a set of data points

and the required number of clusters k, k is specified by the user. The k-means algorithm repeatedly divides the data into k clusters according to a certain distance function.

Metrics: silhouette_score
The contour coefficient is an evaluation method of clustering performance. When silhouette_score is closer to 1, it means that the clustering model's performance is better.

Key Parameter: n_clusters (number of clusters)

The code will loop through the n_clusters in the range of [2,16] and pick the one with the biggest silhouette_score. After the test, n_clusters = 11 has the best silhouette score (0.2029), so I select n_clusters =11 for customer data.

Implementation Process Detail

I used the KMeans function from sklearn.cluster package to create and train the model. The sklearn.metrics package was imported to calculate the silhouette score, which is used to determine the parameter n_clusters. The model's input data is the Udacity_CUSTOMERS_052018.csv file after data processing stage 1 to 8 listed above.

3.2.2 Customer Response Predicting Model

Algorithms: SVM

Solving the classification and regression problems of high-dimensional features is highly effective, and it still has a good effect when the feature dimension is greater than the number of samples. Many kernel functions can be used so that it can be very flexible to solve various nonlinear classification regression problems. It is suitable to deal with skewed data.

Metrics: AUC score

The Kaggle competition scoreboard has the AUC scores from more than 200 teams. Those scores can be used as the benchmark for measuring the model's performance.

Key Parameter: kernel, class_weight, C

Potential complications:

For the SVM model, there is no universal solution to nonlinear problems, and sometimes it is difficult to find a suitable kernel function; SVM is sensitive to missing values, simply replace the missing value with the average value maybe not enough.

The SMOTE process will randomly create the 'fake' positive response customer for our data. The default 'k_neighbors' parameter for SMOTE was 5, which means this fake customer will be created based on its 5 neighbors. If too many fake customers were created, it will harm our model, because more unreasonable fake customers will be created by SMOTE.

The RandomUnderSampler process was used to under-sampling the majority class of raw data. If too many records were ignored by under-sampling, some valuable information from the majority class will also be ignored. This will downgrade our model's performance.

Implementation Process Detail:

Introduced SVM from sklearn to create the SVM model. Chose rbf as kernel function and class_weight = {1:10}, set probability = True to generate possibility as return. The input data is the raw dataset after the 10 data processing steps. Seventy percent of training data used to train the model and the other thirty percent data used for testing model performance.

3.3 Refinement

The refinement process for the K-means model was straightforward, the only parameter to consider is the number of groups. It can be easily determined by silhouette_score. Here I will focus on the SVM model refinement process.

Below is the four major steps refinement I did for the customer response predicting model, followed with detailed explanations how each step being implemented.

1. NN Model Switched to SVM
2. Kernel Selection
3. Penalty parameter C and class_weight.
4. SMOTE strategy parameter

Customer Response Predict Model

Initial try with a neural network model. (parameters: 50 input features, 100 hidden_dim, two hidden layers, epoch 70, all the 366 columns from raw dataset go through the data process and generate 50 input features) The AUC score for the training set is 0.989, for the test set is only 0.492.

The second try with the neural network model. (parameters: 50 input features, 100 hidden_dim, two hidden layers, epoch 70, 114 selected columns from raw dataset go through the data process and generate 50 input features, introduced SMOTE and Under Sampling) The AUC score for the training set is 0.9895, test set 0.9753, but 0.508 for Kaggle competition score. Means my SMOTE process may create too many unreasonable fake records. (SMOTE process was implemented before train/test split, so fake records will be included in the test set.) The under-sampling process may ignore too much useful information.

The third try I changed the sampling strategy for SMOTE and RandomUnderSampler, to create fewer fake records and keep more information from the under-sampling process. Kaggle's competition score is 0.51209, improved a little.

After tried multiple times with the neural network method. I noticed that my test set performance is good (0.97), but the Kaggle competition score is low (0.512). I believe this is because my test data contains the unreasonable fake value from the SMOTE process, those fake value improved

my AUC score for test data. The high imbalance ratio is still a problem for my model, so I decided to try for other classification models. (Because the final model did not choose the neural network, so the neural network code was not included in my submission file.)

I decided to use SVM for this extremely skewed data.

Initial try with SMOTE/ RandomUnderSampler parameter 0.1, 0.2; linear kernel; class_weight= balanced. The test set AUC was 0.572, train set 0.633, Kaggle dataset was 0.54375. After multiple attempts, still cannot improve the AUC score of the training set. It stopped around 0.65. I assume the linear kernel may not be a good fit for our problem. I decided to try with 'rbf' kernel, which has a much better training set AUC score.

Tuning regularization parameter C:

For 'rbf' kernel parameter tuning, the SMOTE/ RandomUnderSampler sample strategy was set to 0.0126. (means the original data will not be over-sampling or under-sampling.) class_weight = {1:10}, C=1. The training set AUC was 0.97, the test set was 0.47. This is an overfitting problem. I decided to decrease the regularization parameter C. After multiple attempts, C =0.95 provide me with the best AUC score. The test set AUC was improved to 0.49.

Tuning class_weight:

Tried {1:5}, {1:10}, {1:15}, {1:30}, {1:50}, {1:100}. The test set AUC score did not improve significantly. Finally, I chose {1:10} for the model.

Tuning SMOTE sample strategy:

After multiple attempts, I notice the model was sensitive to the chosen sample strategy parameters. My test for SMOTE parameters starts from 0.0126 to 0.1, 0.2. The more fake records I created; the Kaggle dataset AUC improved from 0.528 to 0.5509 (when sample strategy = 0.0126*1.5) and then drop down to 0.544 (when sample strategy = 0.0126*3).

After many times of testing, the final model parameters were selected, you can find it from the following Model Evaluation and Validation section.

4 Results

4.1 Model Evaluation and Validation

1.Customer Segmentation report:

The K-means model was selected to create the customer segmentation report. The silhouette_score was used to determine the group number for K-means. The final model divides customers into 11 groups and provide the 11 centroids which clearly shows the characteristic of different customer segmentations. The results found from the model can be trusted.

2.Customer Response Predicting Model

The final model was an SVM model with the following parameters:

SMOTE strategy parameter: 0.0126*1.5

RandomUnderSampler strategy parameter: 0.0126*1.5

Kernel: rbf

class_weight = {1:10}

C=0.95

The final model AUC performance:

Test set: 0.7397871

Train set: 0.9729815

Kaggle dataset: 0.55085

The detailed reason for how the final model was derived can be found from the above 3.3 Refinement section. The final model has some predicting ability for unseen data, but it is limited. This is my first time dealing with a skewed dataset, I do not have a good knowledge reserve for this skewed dataset challenge. Consider the timeline pressure and my knowledge reserve, this is the best result I can get for now.

4.2 Justification

Here is the link to the Kaggle competition scoreboard page. My AUC score was 0.55647. I believe there is a lot of things I can improve for my model. For example, how to deal with the extremely skewed dataset. This is my first time dealing with a skewed dataset, I do not have a good knowledge reserve for this skewed dataset challenge. In the current situation, this is the best result I can get.

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

5 Conclusion

5.1 Free-Form Visualization

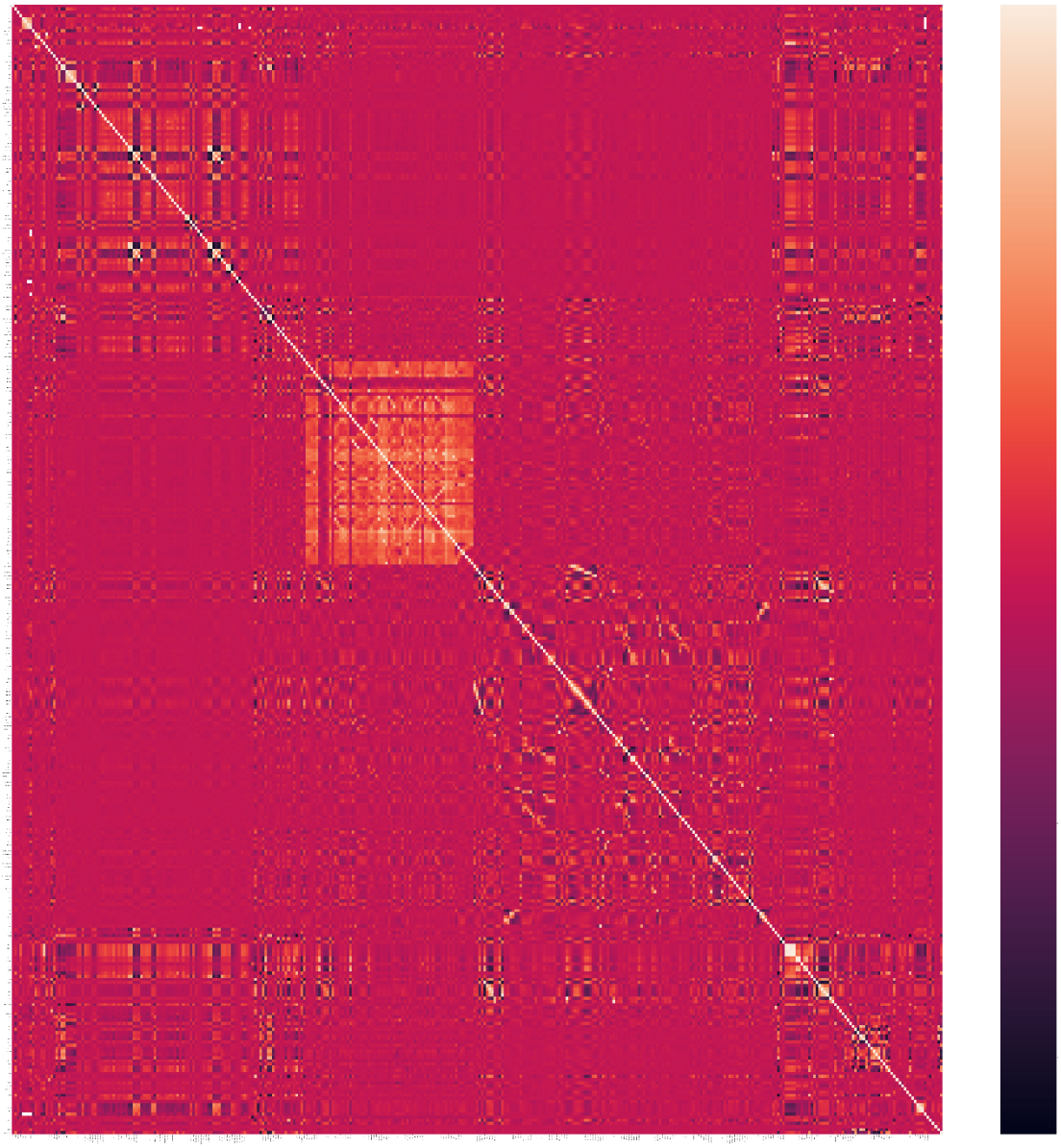

Figure 4 – Correlation Heatmap

The provided raw dataset is a high dimension, high missing rate dataset. It has 366 dimensions and the correlations between those columns are complicated. It makes the data cleaning process challenging. The above picture shows the correlation among the 366 columns. The lighter color means more correlations. We can see that the heatmap color was almost the same light red, which means most of the columns are correlated. That is why I looked through the provide metadata and manually picked 114 columns which are less correlated with each other.

5.2 Reflection

The project includes two parts, the customer segmentation reports and the customer response predicting model. One of the challenges for this project is data cleaning. To deal with this high dimension, high missing rate, skewed raw dataset, the ten data processing steps were implemented. (Please refer to Methodology section for detail). The K-means unsupervised learning algorithm was introduced to generate the customer segmentation report.

For the customer response predicting model: The neural network model was tried first, but the performance was not acceptable (AUC 0.512). Then I switched to the SVM model. After trying multiple time with linear kernel, noticed that the linear kernel training dataset could not generate a high AUC score. (stop around 0.65). Then I switched to Radial Basis Function kernel. The training dataset AUC score was improved a lot (around 0.97), but the test dataset performance was bad (start at 0.49). To solve this overfitting problem, more hyperparameter tuning processes were tried. After tuning with C, class_weight, oversampling parameter (from SMOTE process), the final model Kaggle dataset AUC was improved to 0.55. I believe there is a lot I can improve for my model to get a better result. The extreme skewed dataset is a challenge to me. This is my first time to deal with such a skewed dataset. In the current situation, this is the best result I can give.

5.3 Improvement

During the data cleaning process, the missing value was simply replaced by the average value of that column. Because the SVM model is sensitive to the missing value. If more advanced technics can be introduced (for example Multiple Imputation).

The high imbalance ratio (almost 1/100) caused a lot of trouble for my model. There are only 532 positive responses compare to 42430 negative responses in the training dataset. According to the customer segmentation report which divides customers into 11 groups, I have the reason to believe the 532 positive responses were also scattered around many groups. This makes the skewed dataset more challenging to deal with. Simply using the over-sampling (SMOTE) and under-sampling technic did not solve the problem very well. If more advanced technic can be introduced to deal with the shewed dataset, the model result might be improved.

6 REFERENCE

Here is the list of resource I used to learn from.

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.auc.html