Arvato Financial Services

Machine Learning Engineer Nanodegree

Capstone Project

06 Dec 2021

Contents

Figures

Project Overview

In this project, I analyzed the demographics data of customer of a mail-order sales company in Germany, comparing it against demographics information for the general population. Unsupervised learning methods are used to implement customer segmentation, identifying the parts of population that best describe the core customer base. After identifying the potential customers, I created a supervised classification model to predict which individual would be likely to respond to a marketing campaign. The model was then used to predict an unlabeled dataset, and submitted to Kaggle. Its performance was evaluated by AUC score.

This report was split into two parts. The first part was tailored to Customer Segmentation. In this part, I discussed the problem statement, methods of data pre-processing, modelling, and hyperparameter tuning, as well as key analysis.

The second part of the report was dedicated to Market Prediction towards a marketing campaign. I discussed the problem statement, metrics, methods of data pre-processing, modelling, and refinement, together with key analysis, and conclusions.

1. Customer Segmentation

1.1 Project Introduction

"Customer segmentation is the process of dividing customers into groups based on common characteristics so companies can market to each group effectively and appropriately."[1] In essence, our task was to assign cluster labels to each individual, and give description or evidence on the common characteristics. In this project, I conducted unsupervised learning techniques to general population, and customer group. The portions of each cluster in each group shed light on the favorite and least favorite customers by this German company.

1.2 Problem Statement

- How to segment customers, and what are the features in each cluster?
- Identify the difference or similarity of population and customer data?
- What features stand out for the segmentation?
- Which cluster is overrepresented in general population group compared to customer group?

[1] "Customer Segmentation" by shopify.

- Which cluster is underrepresented in general population group to customer group?

1.3 Metrics

Unlike supervised learning where we have ground truth to evaluate model performance, clustering analysis do not have a solid evaluation metric to evaluate the outcome of different clustering algorithms. [1] The predicted clusters can be used in other prediction models, where we can further evaluate the effectiveness of segmentation using downstream model's evaluation metrics. In K means algorithms, we need to supply K as input. In this project, two metrics were used to select the optimal K: Elbow method by Inertia, and Silhouette Scores.

Inertia is the mean squared distance between each instance and its closet centroid. The Elbow method means plotting the inertia change against number of K and choosing K where the slope inertia started to decrease or stabilized, shown in Figure 7.

Silhouette core is the mean silhouette coefficient over all the instances, ranging from [-1, 1]. A coefficient closes to 1 means that the instance is well inside its own cluster and far from other clusters, while 0 means it is close to a cluster boundary, and finally, -1 means that instance may have been assigned to the wrong cluster.

1.4 Analysis

Data Exploration

Customer Group: 191 652 persons (rows) x 369 features (columns)

General Population: 891 211 persons (rows) x 366 features (columns)

Customer data have extra three features more than General Population:

```
In [6]: customer_info = customers.columns.difference(azdias.columns)
        customer_info
Out[6]: Index(['CUSTOMER_GROUP', 'ONLINE_PURCHASE', 'PRODUCT_GROUP'], dtype='object')
```

Exploratory Visualization

- Portion of missing value in columns in General Population and Customer Group

[1] "K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks" by Imad Dabbura.

The top left boxplot shows 10 outliers as long whiskers after the upper limit, indicating 10 features having enormously large portion of missing values. Similarly, Customer data also had similar fashion with a few large outliers.

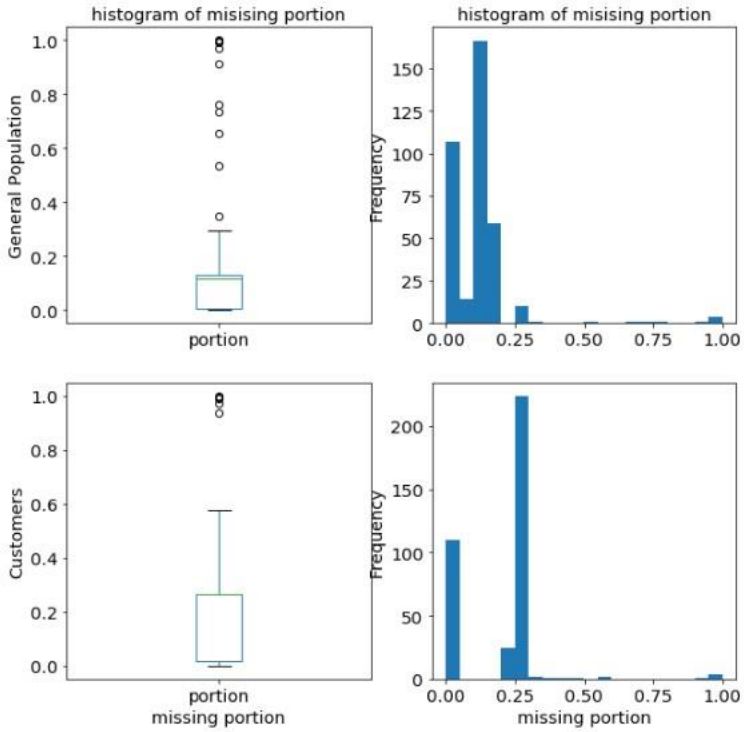


Figure 1 Missing Data in Column Boxplot and Histogram

- Portion of missing values in rows

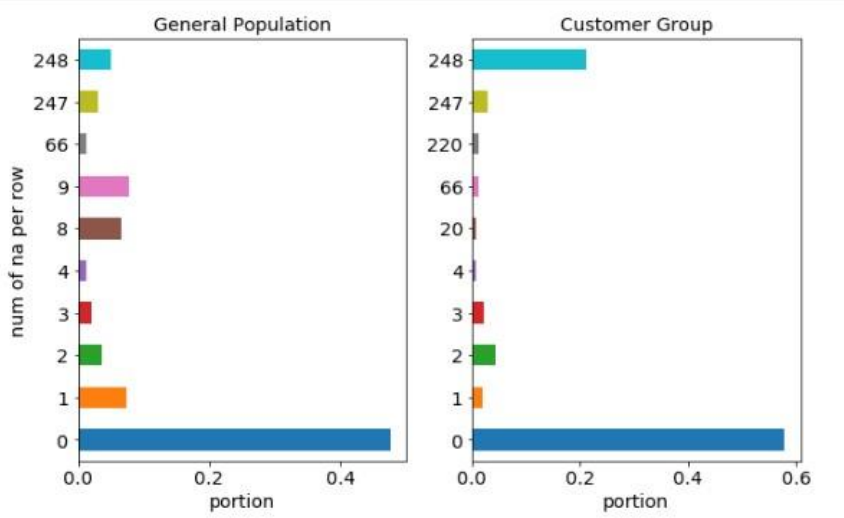


Figure 2 Missing Data in Rows Bar Plot

In Customer Group, almost 30% of rows had 248 features with missing values.

- Comparison of feature distribution in high missing portion group and low missing value portion group

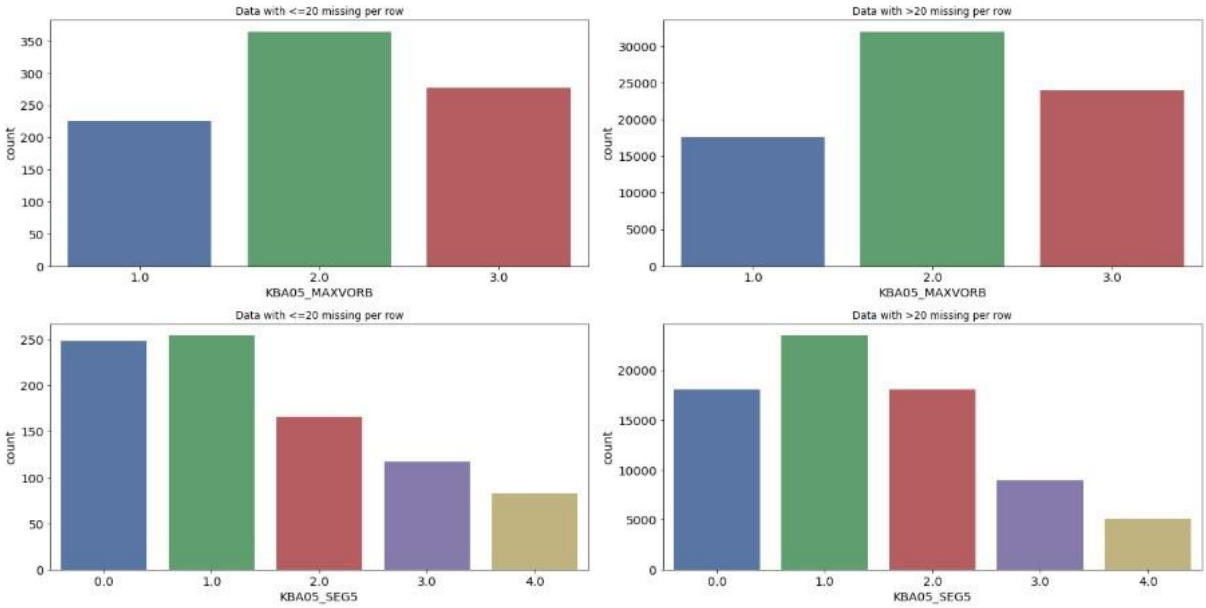


Figure 3 Count Plot

Majority of features in the high portion group had distribution similar to less portion group.

Algorithms and Techniques

- Correlation Analysis

Machine learning algorithms generally don't do well in highly correlated features. Therefore, I generated the correlation table of each possible pairs, setting a threshold to exclude highly collinearly related features.

- PCA

Another way to reduced feature space was Principle Component Analysis, which captured the largest amount of variances in the data by pre-defined number of components. In PCA algorithms, the number of components needed to be provided. A rule of thumb was the threshold where 0.95 cumulative variances are explained by the total number of components.

- K means

K means is a popular method capable of clustering data quickly and efficiently. Note that you have to specify the number of clusters k tat the algorithm must find. Each instance will be assigned to one of the predefined clusters. For each centroid, we calculate the average feature vector of the examples labelled with it. The average feature vectors become the new locations of the centroids. It takes a few iterations to find the optimal cluster centroids.

1.5 Methodology

Data Preprocessing

- Sampling a small portion of data from general population group

  As both datasets were quite large, it occupied a lot of memory and slowed down each modelling step heavily. I shuffled each dataset, and randomly sample 30% of data from each dataset on which to train K-means algorithm.

- Standardizing missing values to 'NP.NAN' format

  Part of the missing values were encoded in 0, -1, X, XX value, while some had 'NaN' value. I made a summary of the missing value code table, and convert them, together with 'NaN' into 'np.nan' format.

	missing_or_unknown	Description
AGER_TYP	[-1]	best-ager typology
ALTERSKATEGORIE_GROB	[-1, 0]	age through prename analysis
ALTER_HH	[0]	main age within the household
ANREDE_KZ	[-1, 0]	gender
BALLRAUM	[-1]	distance to the next metropole

Table 1: Missing Code Example

- Dropping rows and columns with high portion of missing values

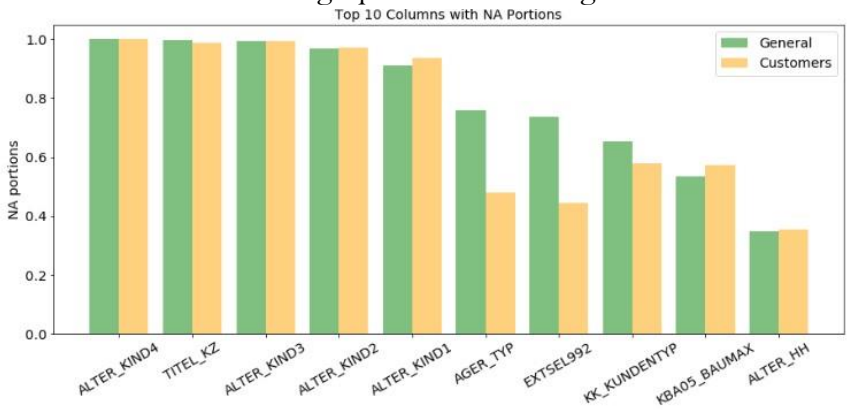


Figure 4 Missing Data in Column Bar Plot

In particular, the portion in 'AGER_TYP' and 'EXTSEL992' were lower in Customer data than in General Population. Nonetheless, these two features carried over 40% of missing values. I discarded these 10 features from modelling.

I also discarded rows with number of missing values greater than 20, except a few cases where its response variable was positive, to retain as much as positive instances to tackle imbalanced dataset problem.

- Cleaned mix type features

A few features such as 'CAMEO_DEU_2015' and 'PRAEGENDE_JUGENDJAHRE' had mixed information, or having wrong data types. For example, 'PRAEGENDE_JUGENDJAHRE' had three levels of meaning in one code, e.g. 2: 40ies - reconstruction years (Avantgarde, O+W). I split this feature into two separate features, each represent time and Avantgarde or not respectively. I also cleaned up 'CAMEO_DEU_2015' data type from object to ordinal numerical label.

- Imputing missing values with median numerical value

Instead of choosing the mean as imputing strategy, I chose median value to represent not only the normally distributed features, but also those skewed, to avoid under or over representation problem. Another strategy would be choosing the mode, the most frequent value.

- Feature Scaling

Features with different scales are likely to be misinterpreted by model. Feature with large range would take away the importance of the features with smaller range. All features are represented in Z scores.

Implementation

- Correlation Analysis

  I ran through a correlation analysis on all the features in General Population data and dropped 132 features that had over 0.6 correlation with other features.

- Principle Component Analysis

  To choose the number of components, I plotted the cumulative explained variances along with the number of components, and chose 181 components corresponding to 0.95 cumulative explained variances, as showed by below figure:

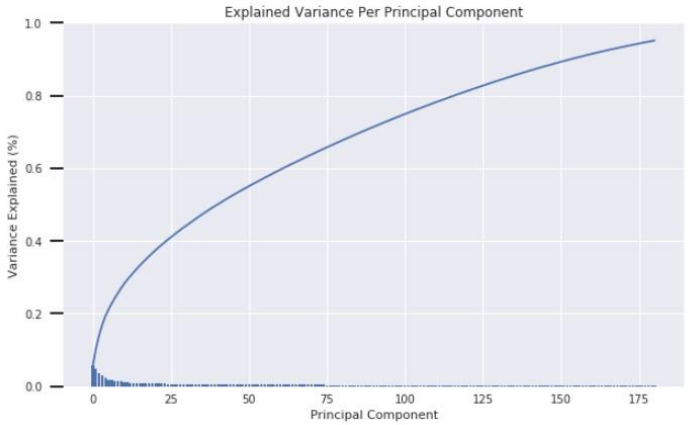


Figure 5 PCA Plot

Each component had weighting scores of all original features, to represent their contribution in the component. Below figure shows the first component and top 10 features with highest weights in this component.

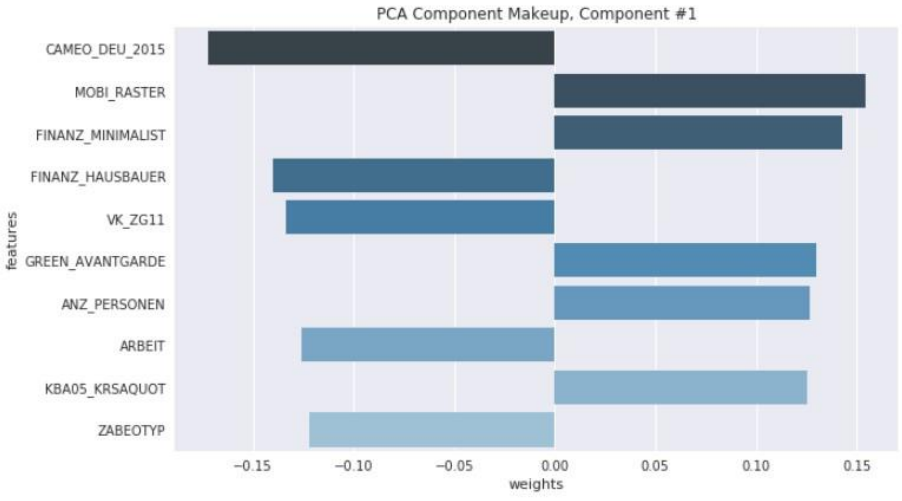
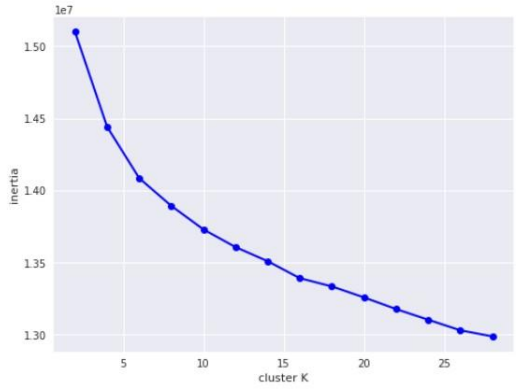


Figure 6 PCA Component Makeup

- Clustering

  Although the elbow shape was not immediately apparent. At K >10, the slope of inertia started to decrease. I chose K=10 as optimal clusters. Similarly, silhouette scores became stabilized after K=10.



Score Figure 7 K means Inertia

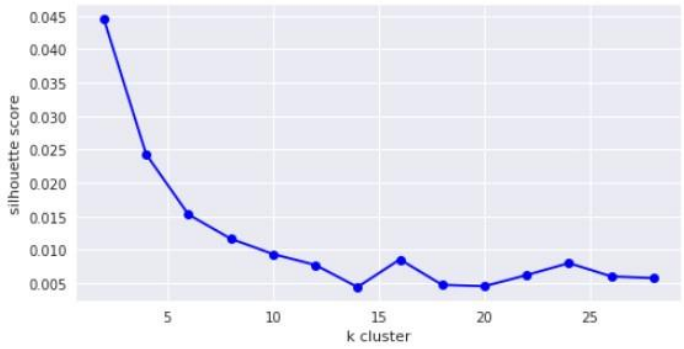


Figure 8 K Means Silhouette

1.6 Results

Model Evaluation and Validation

Assume the General Population and Customer Group were from the same population, the portion of each cluster in the data should be similar. If any segments of the population that are interested in the company's products, then we should see a mismatch in these two data. Figure 9 shows the cluster portion in each general and customer group.

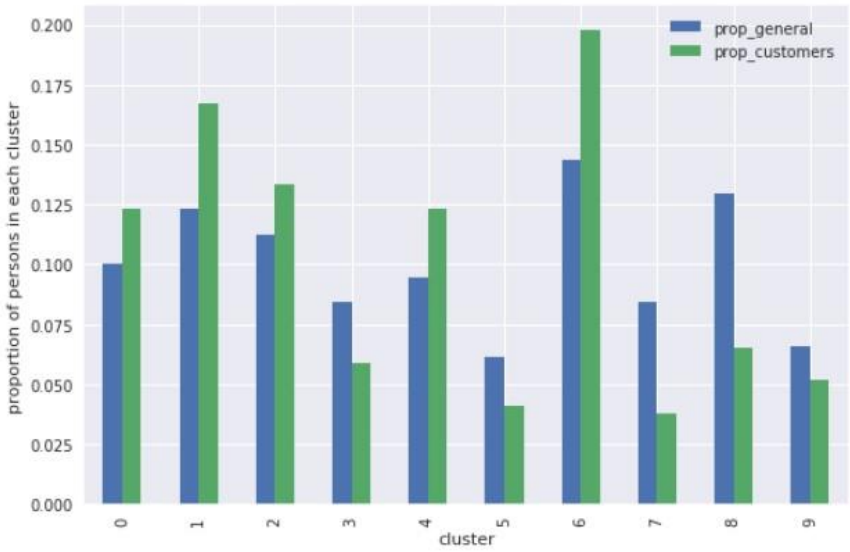


Figure 9 Cluster Portion in Two Groups

Clusters 0, 1, 2, 4, 6 were overrepresented in Customer data compared to General data. The portion of people in these five clusters were significantly higher than those in general data. This implied they were the potential customers on company's products. These would be the company's interest to convert them into customers. On the other hand, cluster 3, 5, 7, 8, 9 were underrepresented in Customer data compared to General data, indicating that they were less likely to be converted. To verify the differences between segments, we could revert to the original features of each segments for comparison.

Figure 10 shows the First 10 Principle Component Centroid Score in each cluster as heatmap. I then selected one overrepresented segment and one underrepresented segment to identify the prominent features.

Justification

Overrepresented – Cluster 0

Cluster 0 occupied 0.1 portion in General Population, while this number was 0.125 in Customer data. The most prominent Principle components were PCA 1, and PCA 3. PCA 3 was primarily affected by 'QTP_TYP_4' negatively, and 'ALTERSKATEGORE_FEIN' positively.

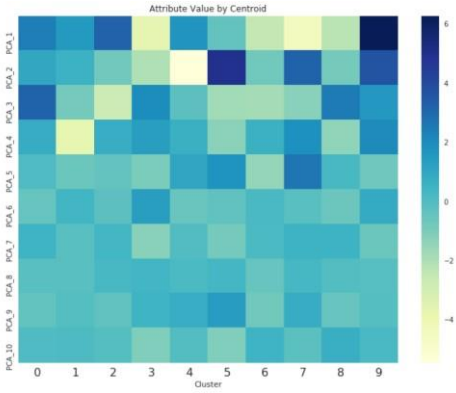


Figure 10: Heatmap of PCA in Clusters

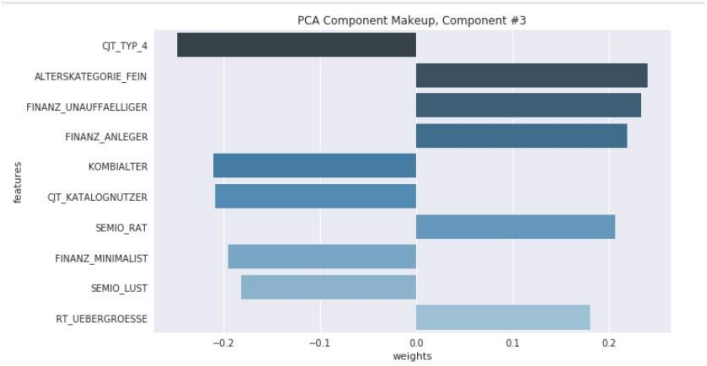


Figure 11: PCA3 in Cluster 0

Finally, from the feature distribution, as showed in Figure 12, we can generate summary statistics of these features compared with general population for this segment. The three selected features showed diverged pattern among Customer and General data. The bar plots on the left side showed the characteristics of Cluster 0 in Customer data. It had a clear diverged skewed direction compared to General population on the right side. If we were to build a recommender system to cluster 0, we would build customer profiles according these prominent features from General Population group.

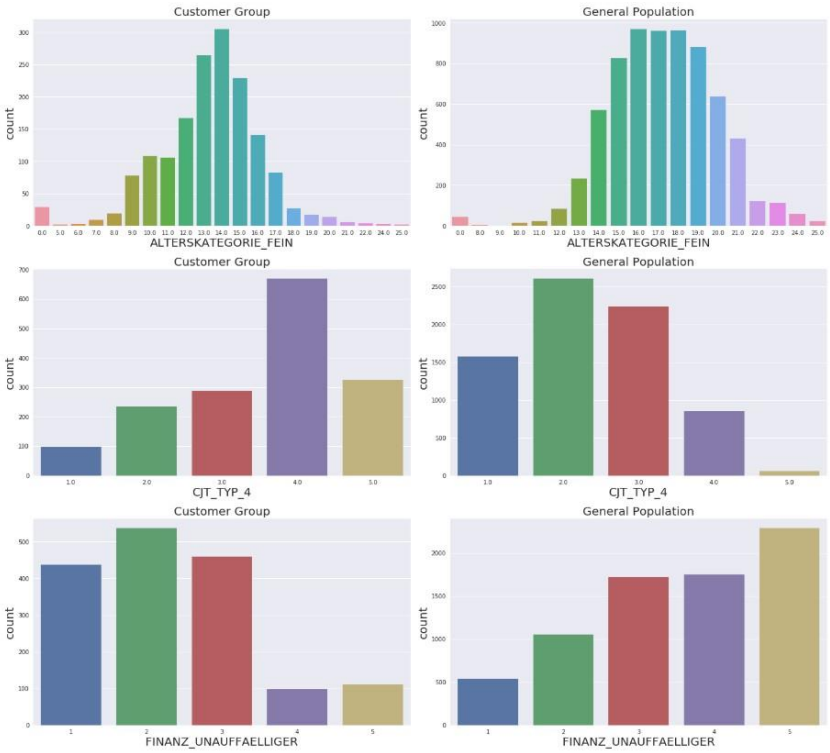


Figure 12: Cluster 0 Feature Analysis

Underrepresented – Cluster 8

Cluster 8 occupied 0.13 portion in General Population, while this number was only 0.06 in Customer data. The most prominent Principal components was also PCA 3. PCA 3 was primarily affected by 'QTP_TYP_4' negatively, and 'ALTERSKATEGORE_FEIN' positively.

In this segment, feature 'CJT_TYP_4' behaved differently in the customer group, compared with segment 0 above. The feature was more uniform distributed in each category, while it was skewed to the lower label in the segment 0.

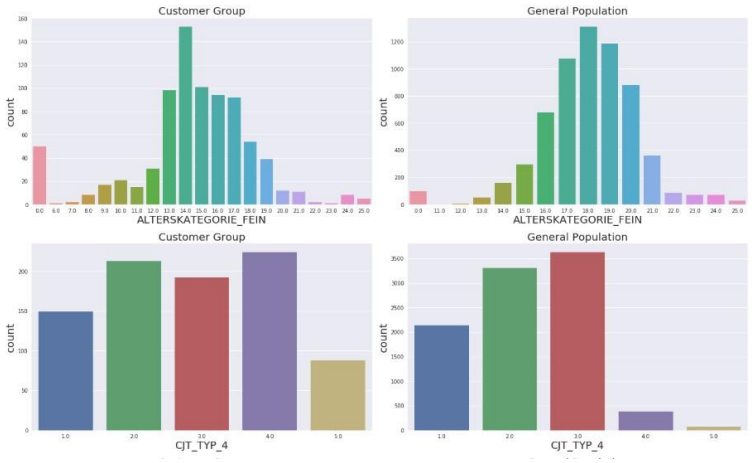


Figure 13 Cluster 8 Feature Distribution

2. Marketing Prediction

2.1 Project Introduction

This is the second part of the report. After segmenting customers and identifying the potential customers, I built a classification model to make prediction if an individual would be likely to convert to customer by responding to the marketing campaign.

2.2 Problem statement

Which individual is likely to be a customer for mail-order business? The potential customers are the target of the marketing campaign.

2.3 Metrics

As the data is imbalanced, meaning that majority of the individuals did not respond to the campaign, Kaggle used AUC score, to evaluate model performance. Ideally, we would like the model to increase true positives rate but control false positive rate.

2.4 Analysis

Data Exploration

The training dataset had 42 982 persons (rows) x 367 (columns).

<class 'pandas.core.frame.DataFrame'> Range Index: 42962 entries, 0 to 42961 Columns: 367 entries, LNR to ALTERSKATEGORIE_GROB dtypes: float64(267), int64(94), object(6) memory usage: 120.3+ MB

Exploratory Visualization

- Missing values in rows

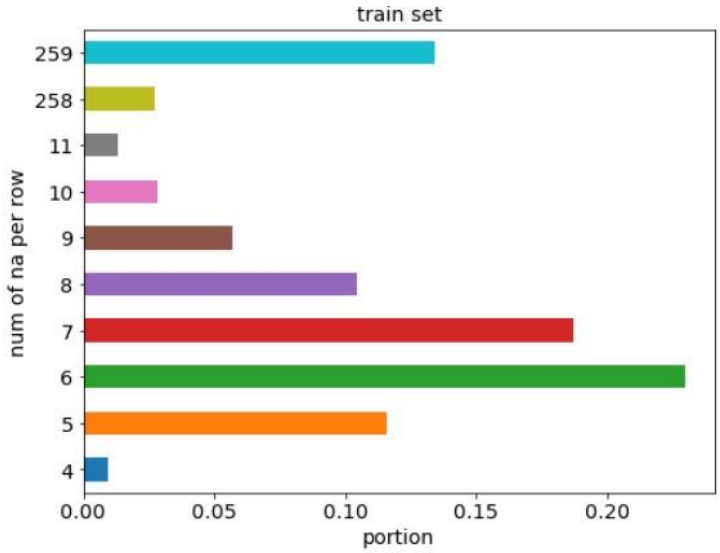


Figure 14: Bar Plot Num of Rows with Missing Values

- Comparison of feature distribution in High Information Group and Low Informative Group

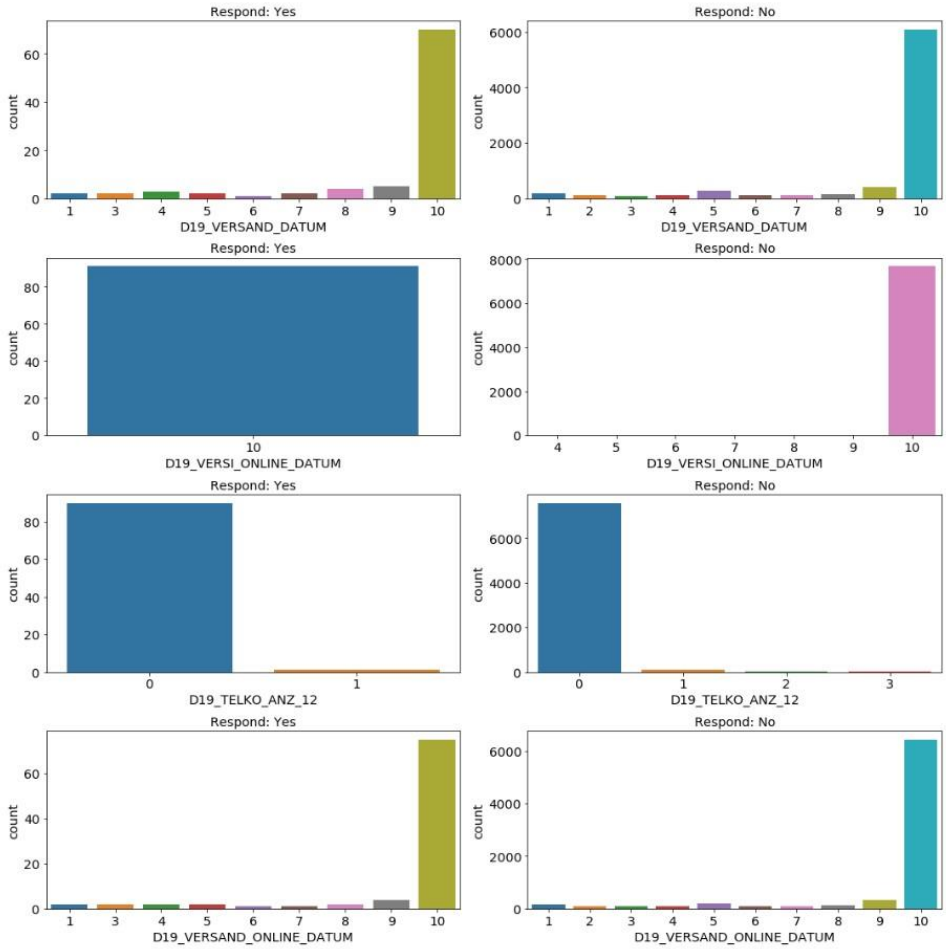


Figure 15: Comparison of feature distribution in High Information Group and Low Informative Group

Algorithms and Techniques

- Principle Component Analysis

  As discussed earlier, this way we can reduce the feature space to reduced level, with selected maximum explained variance.

- Logistic Regression

  Logistic regression is not a regression, but a classification learning algorithm. The name comes from the mathematical formulation, similar to linear regression. In logistic regression, we still want to model y, as a linear function of x, however, with a binary y, this is not straightforward. If we define a negative label as 0 and the positive label as 1, we would just need to find a simple continuous function whose codomain is (0, 1). In such a case, if the value returned by the model for input x is closer to 0, then we assign a

negative label to x; otherwise, the example is labeled as positive. One function that has such property is the standard logistic function, aka. Sigmoid function.

- Random Forest

  Random forest uses a modified tree learning algorithm that inspects, at each split in the learning process, a random subset of the features. The reason for doing this is to avoid the correlation of the trees. Correlation will make bad models more likely to agree, which will hamper the majority of vote in classification problem.

- Gradient Boosting

  Gradient Boosting differs from Random Forest in what the model should learn and how the model should learn. We modified the training set with added residuals gained from training. By computing the residuals, we find how well (or poorly) the target of each training example is predicted by the current model. We then train another tree to fix the errors of the current model and add this new tree to the existing model with some weight. Therefore, each additional tree added to the model partially fixes the errors made by the previous trees until the maximum number M (another hyperparameter) of trees are combined. Gradient boosting usually outperforms random forest in accuracy but, because of its sequential nature, can be significantly slower in training.

- K-Nearest Neighbors

  K-Nearest Neighbors (kNN) is a non-parametric learning algorithm. Unlike other learning algorithms that allow discarding the training data after model is built, kNN keeps all training examples in memory. Wen a new unseen example comes in, the kNN algorithm finds k training examples closet to x and returns the majority label in the case of classification. The closeness of two examples is given by a distance function, such as Euclidean distance, or negative cosine similarity.

- Support Vector Machine

  Support Vector Machine (SVM) is kernel-based algorithm for supervised learning. Its kernel functions can be effective in transforming non-linear data into high dimensions to be separable by linear hyperplane. The basic concept behind SVM is to find the margin between two classes as wide as possible. If the two classes are not linearly separable, lost function is applied to allow data points violate the margin with "price". In this way, margin is maximized, and the error is minimized. In scikit-learn, the kernel function can be chosen between 'linear', 'rbf'(radial basis function), 'polynomial', and 'sigmoid' function, and regularization is chosen by specifying parameter C.

- Imbalanced Data

The negative class ('no') outweighs the positive class ('yes') 99 percent. This problem can be solved by two resampling method: oversampling the minority class by creating synthetic data or undersampling the majority class by dropping instances. Sklearn SMOTE package was used for this implementation .

2.5 Benchmark

Logistic Regression as a simple classification algorithm was used as benchmark. Here I trained a Logistic Regression model with PCA features. AUC score was 0.677.

Figure 16: ROC Curve - Logistic Regression Benchmark

2.6 Methodology

Data Preprocessing

- Clean up rows with large amount of missing value (amount < 20)
- Drop the columns as we defined in Customer Segmentation project, in which I had drop features that were highly correlated or contained large portion of missing values
- Impute missing values with 'median' value of each column
- Scale the features using z-score normalization into standard normal distribution
- Reduce feature space using Principle Component Analysis, to transform raw feature space into reduced space where about 0.95 cumulative variances were captured
- Encode features using catboost encoder which in some way converting the ordinal categorical features in the light of target variable

Implementation

I used PCA features to train four algorithms: Random Forest, kNN, SVM(rbf kernel), and XGBoost.

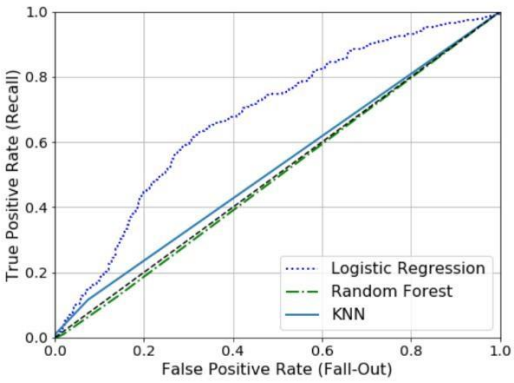


Figure 17: PCA Features ROC Curve

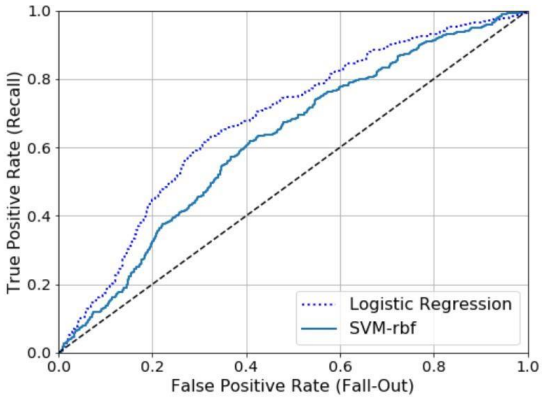
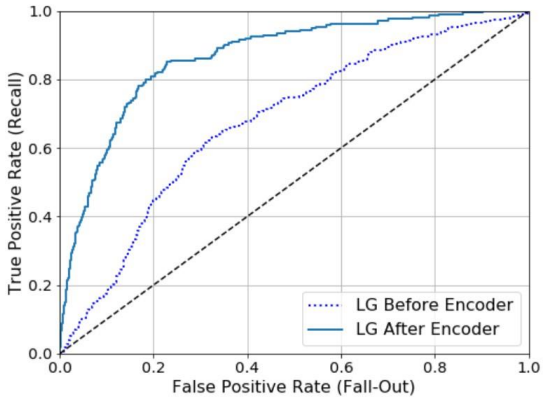


Figure 18: ROC Curve with LG and SVM

From the ROC curves, none of the models outperformed the benchmark model. The SVM model was close to the bench mark but still less than it. XGBoost model had only achieved 0.5 AUC score, basically the same as randomly guessing.

Refinement

Since none of the models outperformed the benchmark model, instead of tuning hyperparameters, I went back to preprocessing the data. This time, instead of using PCA features, I encoded all features using Catboost encoder. Then I retrained the models again. This time all modes had greatly improved AUC score. Even the benchmark model had improved from 0.677 to 0.866. However, all models achieved only about 0.65 AUC score in test data.



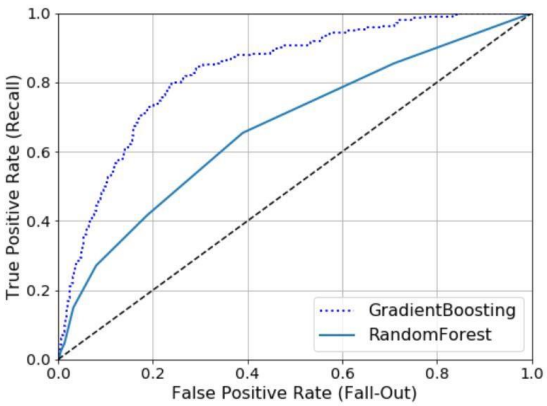


Figure 19: Logistic Regression Trained with Catboost Features

Figure 20: ROC of Gradientboosting and RandomForest on Catboost Features

In addition to feature quality, I also oversampled the minority class (positive response) to ease the imbalanced problem. I retrained the model with this new training data. This time, the best model achieved 0.712 in test data, not very far from its performance in training data.

2.7 Results

Model Evaluation and Validation

Algorithms	PCA Features	Catboost Features	Catboost Features + SMOTE
Logistic Regression	0.677	0.69	NA
kNN	0.53	0.60	NA
Random Forest	0.51	NA	NA
SVM	0.61	0.736	NA
XGBoost	0.5	0.68	NA
Gradient Boosting	NA	0.69	0.76

The best achiever was Gradient Boosting model. This model was trained using catboost encoded features, and modified training set by oversampling minority class.

Analysis

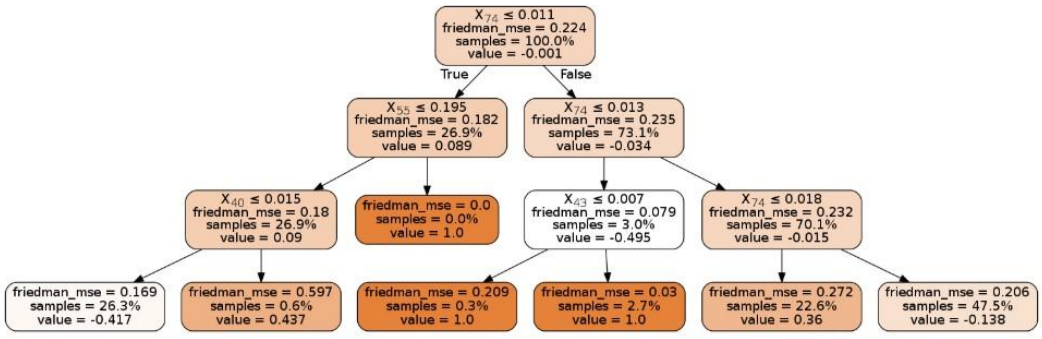


Figure 21: Visualization of Tree Model

I randomly chose a single tree from the fitted model to investigate on the split. In this example, the corresponding features on the split index is 'KBA05_ALTER1', 'D19_VERSI_ONLINE_DATUM', 'D19_SOZIALES', 'D19_SAMMELARTIKEL'.

As showed in the graph, when a new instance X comes to the model, the first split in this tree estimator is feature number 74, depending on the its value against the threshold, it goes down the three until reach to the leaf node. Interestingly, the predicted class is not directly stated in the box. Instead, we sum up the values at the leaf, if it is positive, the predicted class will be positive. Otherwise, it will be negative.

Conclusion

In this project, I trained a K-means model on general population data, and then used the model to cluster the customer data. 10 clusters were retrieved, among which 5 clusters were overrepresented in customer data in terms of number of people, compared against general population. I also identified the prominent Principle components in each cluster using heatmap. Later on, we reverted to the original features in each segments to analyze the feature distribution in both general and customer data, and I found out the divergent pattern in these features between the two data.

In the market prediction analysis project, I used trained Gradient Boosting classifier on training data and used it to predict test data. Even without changing models, by further cleaning data, selecting the right encoding methods, the model could achieve a better result. With hyperparameter tuning, the model was further improved by it was not significant.

I would need to improve results and parameter logging such as using MLflow, or AWS Sagemaker. These tools could automatically record the training process, model artifacts, and performances. In this way, I could better track the model performance, and organized training methods with clear output.

Reference

Burkov, A. (2019). The hundred-page machine learning book (pp. 70-82). Quebec City, Can.: Andriy Burkov.

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media.

ANALYTICS VIDHYA CONTENT TEAM. (2016). Practical Guide to deal with Imbalanced Classification Problems in R. Retrieved from analyticsvidhya: https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classificationproblems/

DMLC. (2015-2016). Introduction to Boosted Trees. Retrieved from xgboost:

http://xgboost.readthedocs.io/en/latest/model.html

G. Lemaitre, F. Nogueira, D. Oliveira, C. Aridas. (2017). SMOTE. Retrieved from scikit-learn: http://contrib.scikit-learn.org/imbalanced-learn/stable/generated/imblearn.over_sampling.SMOTE.html

scikit-learn developers. (2007-2017). Support Vector Machines. Retrieved from scikit-learn: http://scikitlearn.org/stable/modules/svm.html#svm-kernels

Shopify (2020). Customer Segmentation (2020). Retrieved from Shopify: https://www.shopify.com/encyclopedia/customer-segmentation

Imad Dabbura. (2018). K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Retrieved from Towardsdatascience: https://towardsdatascience.com/k-means-clusteringalgorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a?gi=e3ff68d7cb21