Analysis of Credit Card Dataset

Shoval Frydman and Kfir Salomon

April 2020

Abstract

In this paper we analyze a large data set which summarizes the usage behavior of some active credit card holders during a period of time. The file is at a customer level with several behavioral variables. Our main work consisted of applying unsupervised learning methods to this data set. Our goal was to develop a costumer segmentation to define marketing strategy, achieved by applying some unsupervised techniques and algorithms. The results are included in this paper.

1 Introduction

This paper presents the analysis of a dataset which summarizes the usage behavior of 8950 active credit card holders during the last 6 months. The file is at a costumer level with 18 behavioral variables.

Following is the Data Dictionary for Credit Card dataset:

- **CAST_ID:** Identification of Credit Card holder (Categorical)
- **BALANCE:** Balance amount left in their account to make purchases
- **BALANCE_FREQUENCY:** How frequently the Balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated)
- **PURCHASES:** Amount of purchases done in installment
- **ONEOFF_PURCHASES:** Maximum purchase amount done in one-go
- **INSTALLMENTS_PURCHSES:** Amount of purchase done in installment
- **CASH_ADVANCE:** Cash in advance given by the user
- **PURCHASES_FREQUENCY:** How frequently the Purchases are being made, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)

- **ONEOFF_PURCHASES_FREQUENCY:** How frequently Purchases are happening in one-go (1 = frequently done, 0 = not frequently done)
- **PURCHASES_INSTALLMENTS_FREQUENCY:** How frequently purchases in installments are being done (1 = frequently done, 0 = not frequently done)
- **CASH_ADVANCE_FREQUENCY:** How frequently the cash in advance being paid
- **CASH_ADVANCE_TRX:** Number of Transactions made with "Cash in Advanced"
- **PURCHASES_TRX:** Number of purchase transactions made
- **CREDIT_LIMINT:** Limit of Credit Card for user
- **PAYMENTS:** Amount of Payment done by user
- **MINIMUM_PAYMENTS:** Minimum amount of payments made by user
- **PRC_FULL_PAYMENT:** Percent of full payment paid by user
- **TENURE:** Tenure of credit card service for user

As mentioned before, the goal of this paper is to develop a costumer segmentation to define marketing strategy, meaning we will find groups of users the bank would or wouldn't make profits from (by applying clustering methods).

2 Data Analysis

2.1 Preliminary Data Preparation

In this section we will describe our process of data preparation accompanied by some statistical consideration.

- Throughout this paper we will apply statistical methods require the data samples to be independent, and in our dataset they are, because every sample represents a different person's credit card.
- Second, we will search for missing values. Only 2 features have missing information-MINIMUM_PAYMENTS (3.5% is missing) and CREDIT_LIMIT(0.01% is missing). Because it is in small precentages, we will impute these missing values with the median values, and get a full dataset.
- Finally, we will remove the only identifing feature of our dataset: CUST_ID. This feature contains different values for each sample, and as a result contains no information. This leaves us with a dataset of the shape {samples = 8950 , features = 17}.

2.2 Feature Distribution

Our dataset contains only numerical features because CUST_ID was the only categorical one, and we have already removed it. It is important to mention that any future reference of the dataset will refer to its clean form (the one we have achieved at the end of the previous subsection), unless stated otherwise.

In this subsection we will treat our features by showing their correlation matrix and their distribution.

The correlation between all features is shown in figure 1.

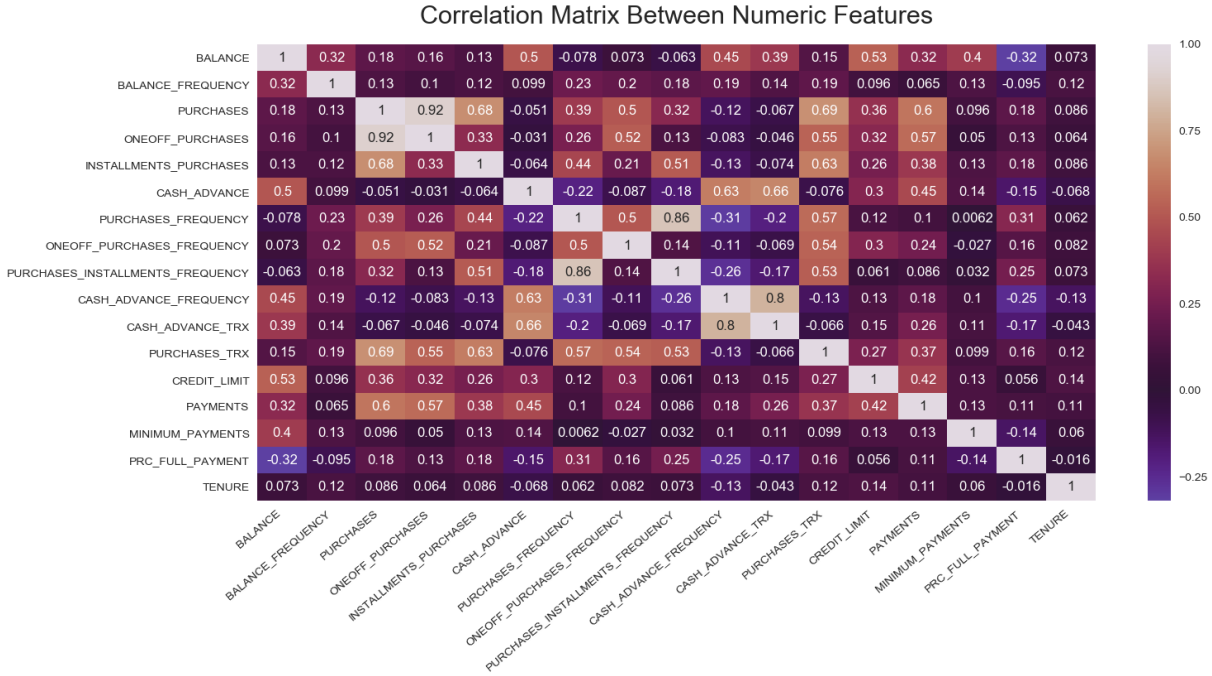

Figure 1: correlation matrix

The features that have the highest correlation are PURCHASES and ONEOFF_PURCHASES, with a correlation cofficient of 0.92. After them we have PURCHASES_INSTALLMENTS_FREQUENCY and PURCHASES_FREQUENCY with 0.86, and CASH_ADVANCE_FREQUENCY and CASH_ADVANCE_TRX with 0.8. The rest of the features have lower correlation- a few of them with 0.5-0.7 and most of them are close to 0, i.e. they are not very correlated. Later, we will try to find connections between them from clustering point of view.

In addition, let us look at the distribution of the features. Figure 2 shows the density of each value. Note that CASH_ADVANCE_TRX, PURCHASES_TRX and TENURE have integer values and the rest of the features have float values.

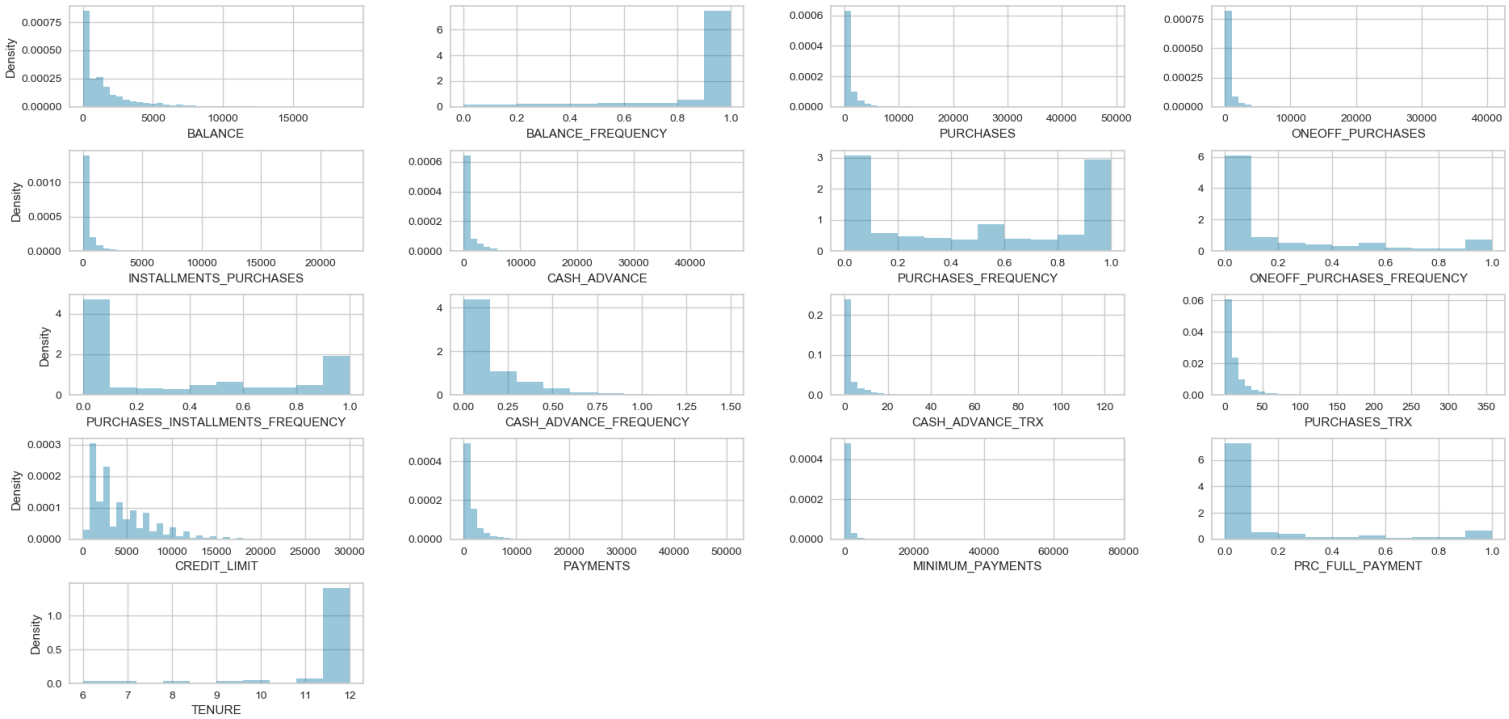

Figure 2: Density of each value, for each feature, in the whole dataset.

2.3 Dealing With Outliers

In statistics, an outlier is a data point that differs significantly from other observations. Plotting box plot (figure 3), we have noticed there are points that seem to be outliers in our dataset. To deal with them, we have decided to use Isolation Forest Algorithm for anomaly detection.

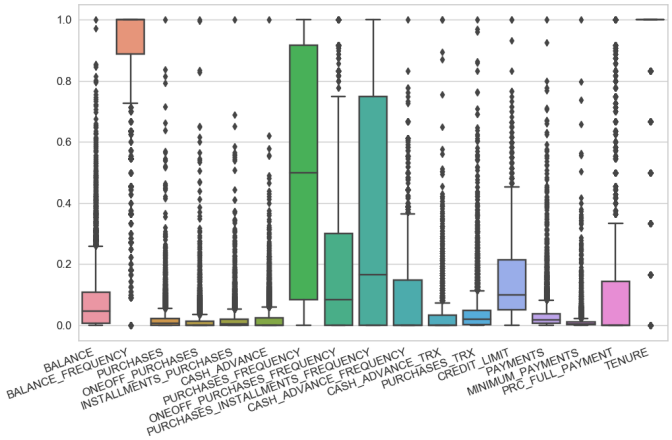

Figure 3: Box plot. Highlighted dots represent outliers

The most common techniques employed for anomaly detection are based on the construction of profile what is "normal": outliers are reported as those instances in the dataset that do not conform to the normal profile. Isolation forest is an unsupervised learning algorithm for anomaly detection that uses a different approach: instead of trying to build a model of normal instances, it explicity isolates anomalous points in the dataset.

After applying the isolation forest algorithm on our dataset, we got 139 instances refrenced as outliers. Figure 4 shows the dataset after min-max normalization and second dimension reduction with PCA (will be explaind in details below), where outliers are colored in red.

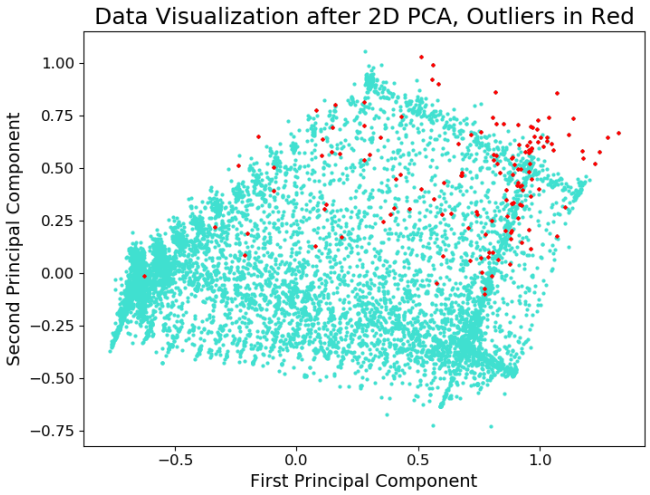

Figure 4: 2D PCA dimension reducion. Outliers are hilghlighted in red

Although finding outliers in our dataset, applying several dimension reduction and clustering methods showed us the dataset functions better with the outliers and gives poor results without them. We can assume that the reason for this behaviour is that every type of customer (in our opinion there are 3 types, will be explained later) has its own outliers, therefore they are different and the dataset behaves very well with them.

2.4 Normalization

To decide in which method to normalize our dataset, we first checked if it is normal with a statistical test- Jarque-Bera test.
This test checks whether the sample data has the skewness and kurtosis matching a normal distribution. If the returned test statistic is far from 0 (also nonnegative), it signals the data doesn't have a normal distribution. We tested it for all our features and got very large numbers for every feature (around 10^4), indicating our data isn't normal. Depending on this result and the performance of clustering, we have decided to normalize the data by min-max normalization.

3 Unsupervised Learning Methods- Dimensionality Reduction and Clustering

In this section we will discuss different dimensionality reduction and clustering methods we have applied.
It is important to mention that all clustering nethods have been applied after dimensionality reduction (gave better results).

3.1 Dimensionality Reduction

We have applied several dimensionality reduction techniques: PCA, Kernel PCA, TSNE and Autoencoder, all of them into 2 and 3 dimensions. In this paper we have decided to show only the 2 demension reduction because clustering methods' performances (measured in several ways, will be explained later) have been drastically better with 2d reduction dimension than 3d, for all applied methods. Figure 5 shows the results.

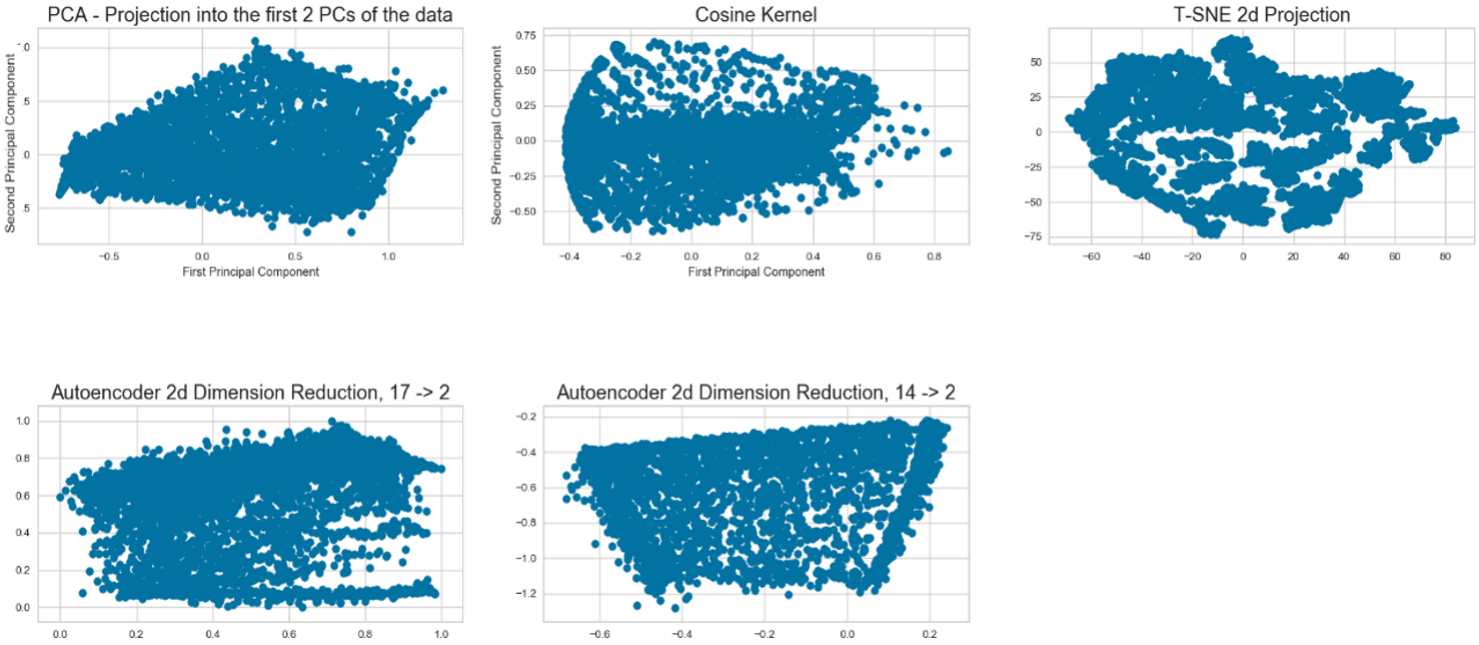

Figure 5: Dimensionality Reduction into 2 dimensions in several methods

1. PCA: Principal Component Analysis was the first method applied on all dataset's samples. We used it to calculate the first 2 principal components of X (the data values), which have explained 63.35% of the variance.

2. Kernel PCA: We applied kernel function in order to map the features to a different feature space. We have tried several kernels (RBF, Polynomial, Cosine), but the cosine

kernel defined as $k(x,y) = \frac{<x,y>}{||x||*||y||}$ gave us the best reults, therefore is presented here. The 2 components have explained 63.36% of the variance, almost identical to regular PCA.

3. T-SNE: t-SNE converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. The dimension reduction was to 2 dimensions, and we set perplexity parameter (related to the number of nearest neighbours) to be 50.

4. Autoencoder 17→2: An autoencoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction- as used here.
The autoencoder we have built is made of 2 parts: an encoder and a decoder. The encoder encodes the data into a requested reduced dimension representation and actually it does the dimension reduction. It has 3 layers: the input layer of size 17, one hidden layer of size 8 and the output layer of size 2- the requested reduced dimension. The decoder decodes the encoded data and brings it back to its original dimension. It also has 3 layers: the input layer of size 2, a hidden layer of size 8 and an output layer of size 17. We have trained it for 7 epochs, with Tanh activation function, Adam optimizer with learning rate of 0.008 and Mean Squered Error (MSE) loss function. To choose these hyperparameters we trained it with 80% of the samples and tested it with the remaining 20%. Then we have trained it with all the samples in order to get the encoded data that is in a lower dimension, in our case 2D. The decoder is being used only for training- to make sure we choose the optimal hyperparameters. For the dimension reduction itself only the encoder is needed.

5. Autoencoder 14→2: Doing a trial in which in every iteration we removed one feature from the data to check how dimension reduction and clustering performe without it, we found out that there are 4 features that without them the performance is better: BALANCE_FREQUENCY, PRC_FULL_PAYMENT, TENURE and BALANCE. Therfore, we have decided to merge them into a one vector representation with an autoencoder, i.e. dimension reduction from 4 features to 1 feature. We have done that with an autoencoder with an encoder with an input layer of size 4, one hidden layer of size 2 and output layer of size 1 (the decoder is built in a reversed structure). This way we've gotten 1 feature instead of 4. We have removed them from the data ,inserted the encoded one, and been left with 14 features. These 14 features we inserted to the autoencoder described in 4 (layers' sizes are 14, 7, 2, in the encoder and reversed for the decoder) and got an encoded repersentation of demension 2D, as can be seen in figure 5.

For clustering we have decided to work only with the PCA and the second autoencoder dimensionallity reduction method for two reasons: first, for both of them we have

gotten the best results (variance explanation and clustering) and second, PCA is a linear method for dimension reduction, while autoencoder is a non-linear one, so we wanted to compare between them.
Notice that when autoencdoer method is mentioned, if not said otherwise we refer to the second autoencoder method.

3.2 Clustering

Looking on the generated figures, we can not immediately notice the number of clusters these points are separated into because they seem very close to each other, both in PCA and autoencoder. Not knowing which method is better be used, we have decided to apply several of them: K-means, GMM, and Agglomerative Hierarchical Clustering.

3.2.1 Methods to Determine Number of Clusters

Before any clustering is made, the number of clusters needs to be determined. In order to choose it, we used both the elbow method and silhouette scores to help us measuring the performance of the clustering.
The elbow method is a heuristic method that looks at the precentage of variance explained as a function of the number of clusters. One should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data. As can be seen in figure 6, acoording to this method for K-means algorithm, we should choose to separate the data into 3 clusters.

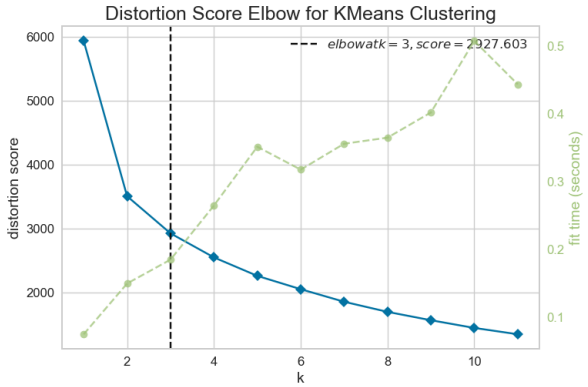

Figure 6: The Elbow Method for K-means clustering algorithm

Another method applyed to determine the number of clusters is calculating average silouhette score for a number of clusters (2-6). Figures 7 and 8 show this for K-means, both with PCA and Autoencoder, respectively.
The silouhette value is a measure of how similar an object is to its own cluster compared to other clusters. It ranges between -1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. As can

be seen in figures 7 and 8, for K-means with both dimensionality reduction methods, one should choose 3 clusters because its average silhoutte score is the highest, with almost no negative specific scores (and if so, very little).

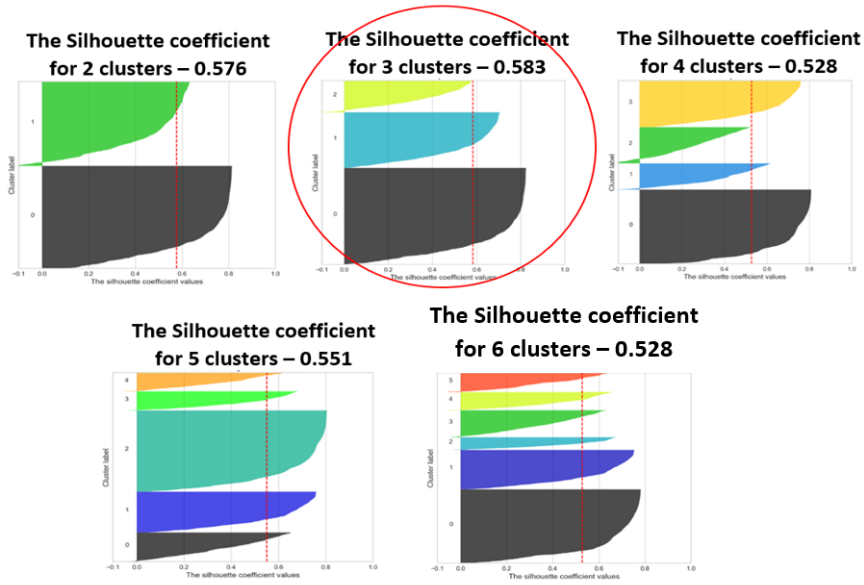

Figure 7: Silhouette Scores for K-means algorithm, visualize with 2D PCA

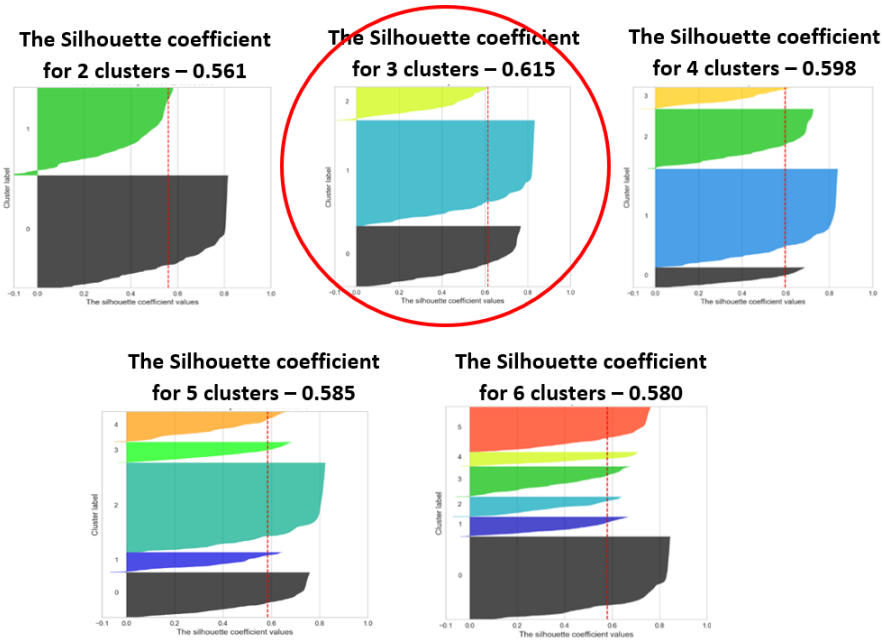

Figure 8: Silhouette Scores for K-means algorithm, visualize with 2D Autoencoder

Although we will not show this here, also for GMM and HC we have gotten that 3 clusters is the best option according to average silouhette scores.

To sum up, for both different dimensionality reduction and clustering methods we got that 3 clusters is the ideal number, thus for all different methods we will separate the data into 3 clusters, using silouhette score and other scores to compare between them.

3.2.2 Measuring Clustering

In order to measure the performance of the clustering, we used 3 scores:
1. Silhouette score: Explained above.
2. Davis Bouldin score: The score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. The minimum score is 0, with lower values indicating better clustering.
3. Calinski Harabasz index: The ratio of the sum of between-clusters dispersion and of inter-cluster dispersion for all clusters (where dispersion is defined as the sum of distances squared). The score is higher when clusters are dense and well seperated, which relates to a standard concept of a cluster.

3.2.3 Applying Clustering Methods

Figures 9 and 10 show the visualization results of K-means, GMM and Agglomerative Hierarchical clustering, with 3 clusters, for both PCA and Autoencoder respectively.
The table in figure 11 summarizes the values of the quality measures for clustering (see 3.2.2 for an explanation).

Kmeans was initialized with a random state of 0. GMM was initialized with cluster means that were generated first by the K-means algorithm. Agglomerative Hierarchical Clustering was initialized with linkage criterion 'complete' and affinity criterion 'euclidean' that gave the best results. The first determines which distance to use between sets of observations where the algorithm will merge the pairs of clusters that minimize this criterion, and the second determines metric used to compute the linkage.

Looking on the K-means results with PCA and Autoencoder representations, despite the different visualization, it can be clearly noticed by eye that the brown cluster of PCA is like the blue cluster of Autoencoder, the blue as the brown and light blue as light blue. It can be noticed as well in Hierarchical Clustering- blue as the brown, brown as the light blue and light blue as the blue.

Looking on the table in figure 11, one can see that the autoencoder representation gave much better results than PCA for all clustering methods and for all measures, therfore we can assume a non-linear model fits better to our dataset.

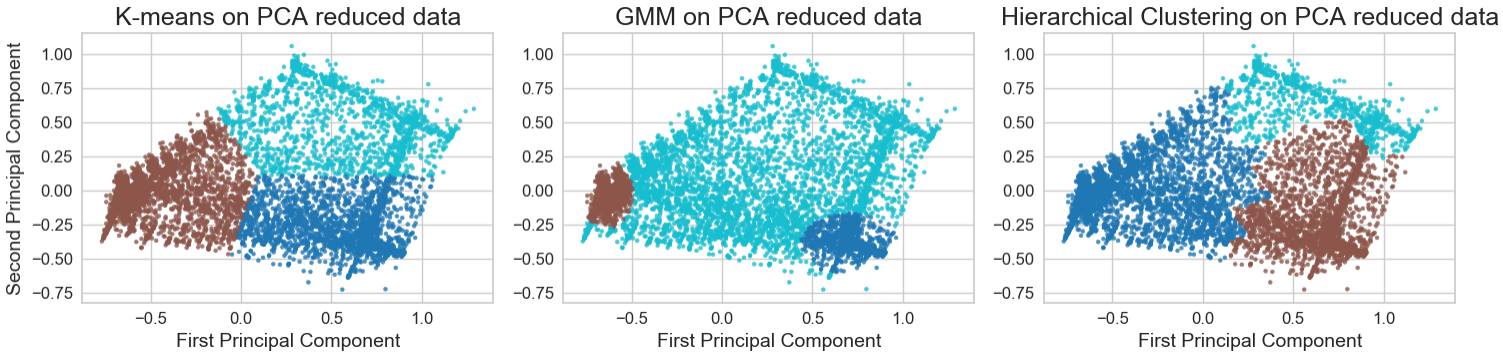

Figure 9: Applying Clustering Methods, PCA

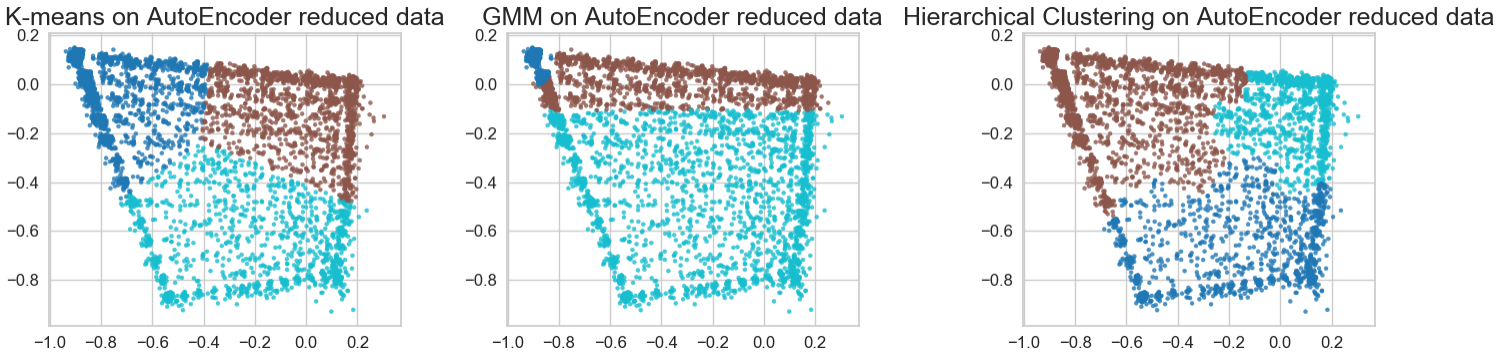

Figure 10: Applying Clustering Methods, AutoEncoder

	Silhouette Score	Davis Bouldin score	Calinski Harabasz index
K-means	**0.583**	0.750	**16403.998**
	0.612	0.705	**17092.344**
GMM	0.283	0.867	5716.695
	0.322	1.278	5230.337
Hierarchical Clustering	0.554	**0.653**	12767.677
	0.574	**0.640**	13386.446

Figure 11: Scores values for clusering. Blue and black for PCA and Autoencoder, respectively. The highlighted ones are the best ones among indices.

As can be seen in the figures, with both representations K-means's performance was the best, both in visualization and quality indices (2 out of 3), as summarized in figure 11. Hierarchical Clustering also gave pretty good results and even gave the best values for Davis Bouldin score, but in the visualization the seperation seems a little odd because parts of the clusters browse into each other. GMM gave very poor results and failed to separate the data.

Depending on all the above, we have decided K-means with an autoencoder dimension reduction representation is best for our data, therefore it is chosen to be the model we will keep working on and get conclusions from.

3.3 Labels

Our dataset has no labels, so measuring the excellence of the clustering is not so obivious. To deal with this problem we have tried several approaches, all very similar- take one feature that seems to have a great influence on the clustering performance and make it into a label by sorting all samples by its value, and devide the samples into 3 clusters, each cluster with the same number of points. The difference between the various approaches is how to choose this feature.

1. First we have tried to take one feature as is from the dataset's features. We chose PURCHASES feature and PURCHASES_FREQUENCY feature that seem to affect greatly on the separation into clusters.
2. Second, we have inserted all 17 features into an autoencoder to get a one vector representation and this created feature make into a label. Note that this method labels the data according to all features, thus both labels and clustering rely on the same features and are very correlated. Therfore, this method is not our favorite, but we still mention it.

Although these ideas sound promising on paper (at least we think they do), they gave no special results, measured by mutal information index. In figure 12 one can see the result for every labeling method, compared to assiging labels randomly to every sample. We measure the quality of the labels with mutal information index (a measure of the mutual dependence between two variables).

	Mutual Information between K-means predictions and our labels	Mutual Information between K-means predictions and random labels
With 'PURCHASES' feature	0.29149	0.0000913
	0.00018	0.0000186
With 'PURCHASES_FREQUENCY' feature	**0.46587**	0.0000619
	0.00010	0.0000120
With 17 features merged into one with an autoencoder	0.00026	0.0000063
	0.00019	0.0000052

Figure 12: Mutal Information values for all labeling methods. Blue and black for PCA and Autoencoder, respectively. The highlighted ones are the best ones among methods.

As can be seen in table 12, for PCA the first approach with PURCHASES_FREQUENCY feature has given the highest value of mutual information. For autoencoder, the second

approach has performed better, but hardly, with a very little difference between all methods. Generally, unexpectedly, dimension reduction with autoencoder in this section has given poor results compared with PCA, although in the previous section we have chosen the autoencoder to be the model that fits best to our data (therefore we assumed it will perform better in this mission). We think this might have happened for several reasons. First, the labels have been made artificialy by us, so they don't necessarily represent the real labelels of our data correctly. Because of this, the labels we have gotten from k-means with autoencoder aren't necessrily poor as one might think. Second, clustering measures (e.g. siluhette score) are a way to evaluate the quality of the clusters, but they need to be taken with a grain of salt. Indeed for autoencoder we have gotten better clustering scores, but not so far from PCA.
In addition, for all labeling methods the random state is worse than our artificial labels.

4 Conclusions

As mentioned in the introduction, our final goal is to develop a costumer segmentation to define marketing strategy. We want to find groups of users the bank would or wouldn't make profits from. In order to do that, we have chosen 6 features that seem to represent our data best: BALANCE, PURCHASES, CASH_ADVANCE, CREDIT_LIMIT, PAYMENTS, MINIMUM_PAYMENTS. We have performed k-means clustering (with 3 clusters) on our data (only with these 6 features). Figure 13 shows a pair plot which represents the connection between each feature and cluster.
Looking on this pairplot, we decided to separate the people to three different types:

1. The Green- The Wealthy:
This group of people has the highest balance, purchases and payments. There is a linear connection between balance and credit limit, i.e. their bank account is stable and thus the bank can give them high credit limit. Between payments and purchases, and purchases and credit limit, there is also a linear connection, indicating that they make lots of purchases (they can because they have high credit limit) but also pay for the bank (high payments). Still, they are stable costumers and do not afraid to spend their money. Therfore, we assume they are the wealthy people.

2. The Red- The Money Borrowers:
This group of people has a small balance, avergae purchases but a lot of payments and pretty high credit limit compared with their very low balance. Looking on figure 13, one can see the connection between balance and credit limit of the red group. Usually, when your bank account is stable, it is likely that the bank would give you a high credit limit. However, for the red group, the credit limit is high althogh their balance is pretty low. Thus, we can assume they take loans from the bank. Looking on the connection between credit limit and purchases, one can see that they do not buy a lot althogh they do have a high credit limit, a fact that strenghtens the claim above. Also, they have low purchases but high payments. Therefore, we assume they are the money borrowers.

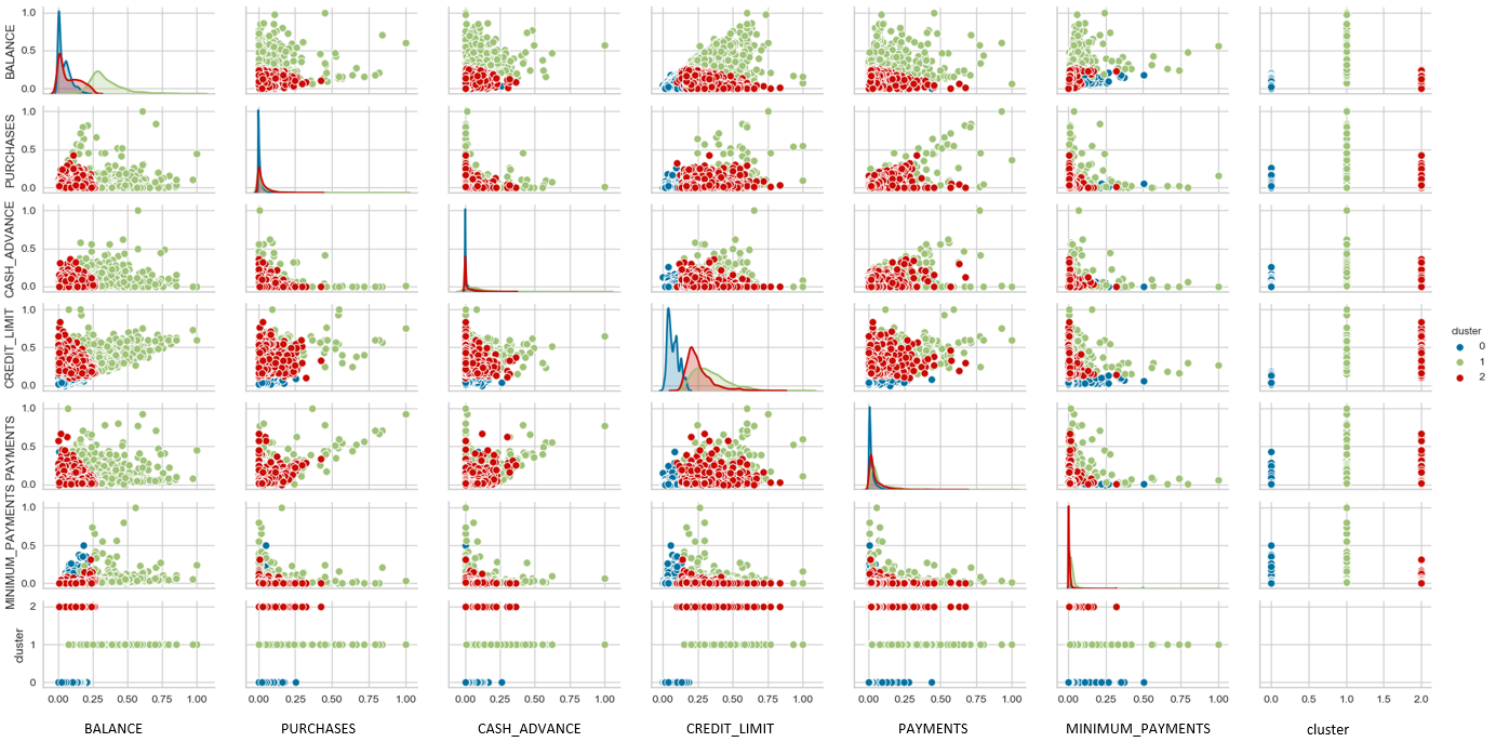

Figure 13: Mutal Information values for all labeling methods. Blue and black for PCA and Autoencoder, respectively. The highlighted ones are the best ones among methods.

3. The Blue- The Poor:
This group of people has the lowest balance, purchases, credit limit and payments. Their credit limit is low as the balance and as the purchases. It seems they do not have a lot of money and therefore can not afford making a lot of purchases. Thus, we assume they are the group of poor people.

In conclusion, the green group's people are worthy customers with minimum risk- they have economic stability, they spend a lot and take payments accordingly. Therfore, we recommend the bank to treat them well and keep them as their clients.
The red group's people are both worthy but risky. Although they take big loans from the bank, they don't have a big budget, and can easily go bunkrupt, and it is not good for the bank. Therfore, we recommend the bank to limit their loans according to their balance.
The blue group's people do not worth for the bank, but usually they are not risky. They do not have a high budget, however they do not take big loans. Still, we do not recommend the bank to give up on them because they are many and separately each one of them is not risky for the bank. As a result, together they will probably be worthy to the bank.