In this article we dive into Customer Segmentation with LLM and experiment with three different methods : Kmeans , K-Prototype and LLM + Kmeans. Also there are three dimensionality reduction techniques mentioned : PCA , t-SNE ,and MCA. The dataset used is from Kaggle and is about Banking Data.

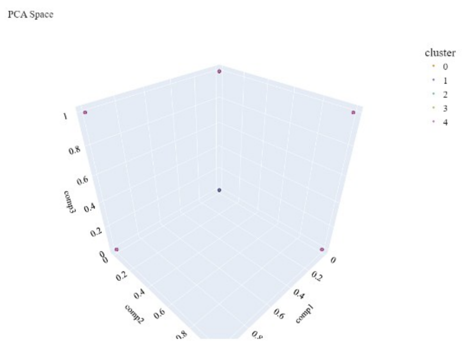
Method 1: Kmeans

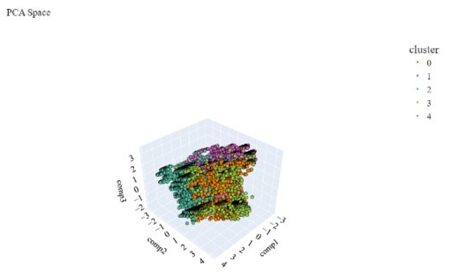
Before implementing the actual method we need to preprocess our data , that is done by One-Hot-Encoding our Data through a pipeline , using ordinal encoding and also using the **PowerTransformer** which applies a power transformation to make the data more Gaussian-like, which can be beneficial for our project. The different Pipelines are combined with ColumnTransformer and after that we create the Final Pipeline which need to be Fitted and Transformed.

Next in the next part of preprocessing we need to detect and handle the Outliers , which is done by the Python Outlier Detection (PyOD) library. Specifically we are using the ECOD method. Basically we find unusual data points with ECOD( ) and then we tarin the ECOD model on our dataset stored in the variable data. After training, we use the predict() method to identify potential outliers in the data. The results (0 for normal data points and 1 for outliers) are stored in the outliers variable. Lastly we add a new column named "outliers" to our data and store the results (0s and 1s) from the outlier detection step in this column. This effectively labels each data point as either an outlier or not. In essence, our code identifies outliers in the data, creates two separate datasets (one with outliers removed and one with outliers included), and then displays the size of each dataset. This separation allows for further analysis and model building with and without the potential influence of outliers.

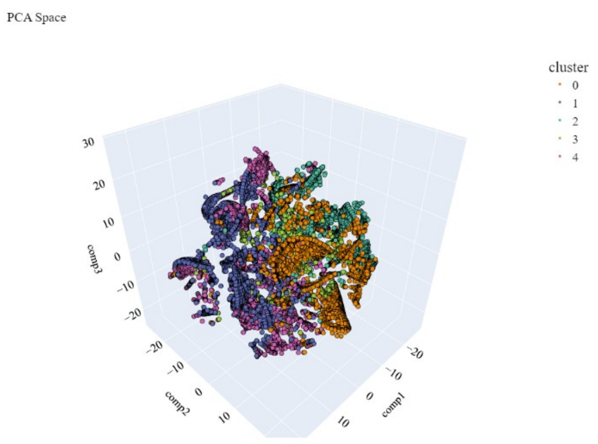
Next , we take advantage of the Elbow Method to decide the number of clusters to use which as a process is the main disadvantage of the Kmeans algorithm. We are calculating the distortion that exists between the points of a cluster and its centroid while trying to obtain the least possible distortion. After excecuting the described code and visually representing the results to help with the decision , we see that k=5 is the best scenario. We also perform Silhouette visualization and use the whole process to save a series of plots. These plots, known as Silhouette plots, are a way to visualize the quality of clustering results. They help assess how well each data point fits within its assigned cluster compared to other clusters. For this we define a function that creates the Silhouette plot and then compute the silhouette scores for each sample.We see that the highest silhouette score is obtained with n_cluster=9 , although after k = 5 the variation in the score is very minimal. That drives us to the conclusion that we should probably use k = 5 or 6 . Now we can finally actually create the Kmeans model with K=5.

The last thing we have to do is to evaluate the model. In order to do that we use different metrics like Davies-Bouldin index, Calinski-Harabasz index, and Silhouette score.This is done easily by using sklearn.metrics and after implementing the different libraries needed we now only have to print the scores. Each metric provides a different perspective on the clustering performance, helping to assess how well the model has grouped the data. Analyzing these scores helps to determine the effectiveness of the chosen clustering algorithm and the number of clusters used. The scores give us mixed opinions , but this may be due to several factors. We’ll now use PCA to reduce dimensionality. We use get_pca_2d(df, predict) to perform PCA to reduce the dimensionality of the data to 2 principal components. After , we do the same with get_pca_3d(df, predict) but for 3 principal components instead of 2 this time.The last thing left is to visualize both results . It can be seen that the clusters have almost no separation between them and there is no clear division. This is in accordance with the information provided by the metrics.





When we apply the PCA method, since it is a linear algorithm, it is not capable of capturing more complex relationships. Luckily there is a method called t-SNE, which is capable of capturing these complex polynomial relationships. This can help us visualize, since with the previous method we have not had much success.



It can be seen that feature age has the greatest predictive power. It can also be seen that cluster number 3 (green) is mainly differentiated by the balance variable.

Finally we must analyze the characteristics of the clusters. This part of the study is what is decisive for the business. After carrying out the analysis in different ways, they converge on the same conclusion: “We need to improve the results”.

Method 2:K-Prototype

Again the first thing we have to do is to Preprocess our data . The reason is that we have numerical values and of course we have to apply modifications , meaning that we have to scale them properly with distributions as close as possible to Gaussian ones.

We use a similar approach to the first method. We create a Pipeline for **Power Transformation ,** then **Apply the Transformation** and then we **Create a Copy and Replace the Columns.**

In essence, the code prepares the data for further analysis by applying a Power Transformation to specific numerical features, aiming to improve their distribution for modeling purposes.

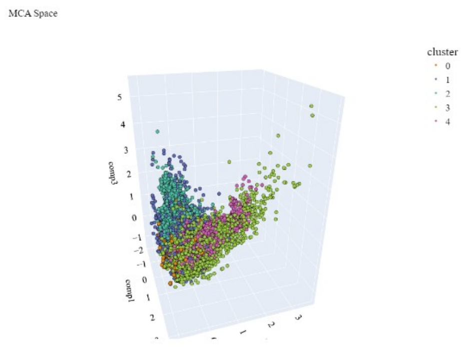
Next we handle the outliers the same way we did in the first method.

Now on the Modeling part we use the Elbow Method. The code iterates through each cluster number in the specified range. What’s next is that it plots and by looking at the plot we see that it suggests the optimal number of clusters for the K-Prototypes algorithm is K = 5.

We now build the model and since this method is similar to the first one we now go straight to evaluating it.

For the evaluation part we will use the MCA method by using the functions from the given code. These functions are designed to perform MCA and reduce the dimensionality of the data to 3 components.

The functions get_MCA_3d and get_MCA_2d primarily perform Multiple Correspondence Analysis (MCA) to reduce the dimensionality of categorical data for visualization and analysis. They create an MCA object, specifying the desired number of components (3 for get_MCA_3d and 2 for get_MCA_2d) and other parameters. Then, they apply the MCA transformation to the input data, resulting in a new DataFrame with reduced dimensions. The column names of this transformed DataFrame are set to "comp1", "comp2" (and "comp3" for the 3D case), representing the principal components. Finally, they add a "cluster" column containing the cluster assignments for each data point before returning the MCA object and the transformed DataFrame. In essence, these functions simplify complex categorical data while preserving important relationships for clustering analysis.



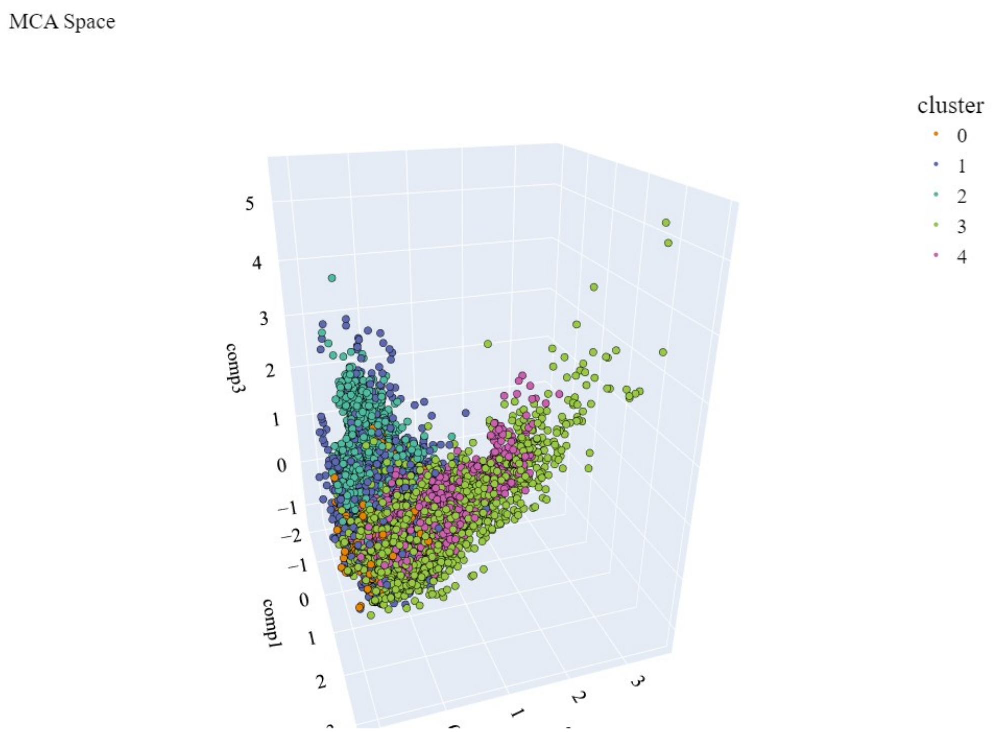
Next , using “mca_3d.eigenvalues_summary” we obtain the valus of the first 3 components to be able to draw conclusions.

We’ll now take these steps:

1. Apply PCA to the dataset to which preprocessing has been performed to transform the categorical variables into numerical ones.

2. Obtain the components of the PCA

3. Make a representation using the PCA components such as the axes and the color of the points to predict the K-Prototype model.



Method 3: LLM +Kmeans

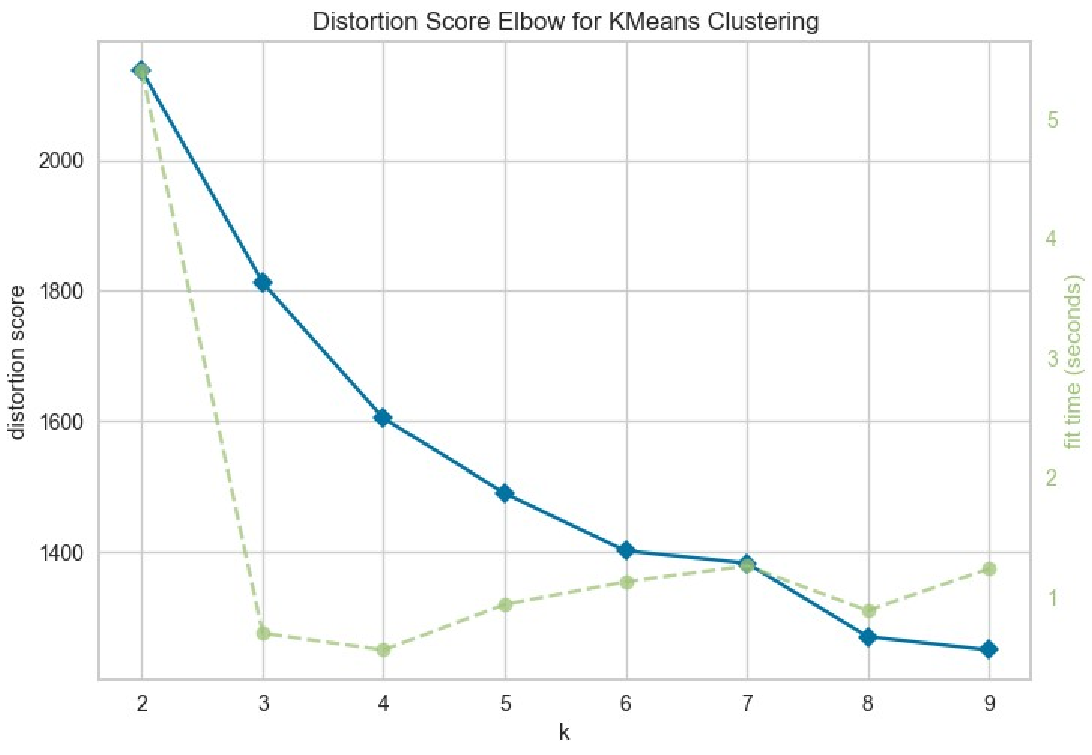
It will help a lot for our method to perform Sentence Embeddings , which is done with the given code. Essentially we encode information of each client and unify it into text that contains all its characteristics. The steos taken are:

1) The text is created for each row, which contains the complete customer/row information. We also store it in a python list for later use.
2) This is when the call to the transformer is made. For this we are going to use the model stored in HuggingFace. This model is specifically trained to perform embedding at the sentence level, unlike Bert’s model, which is focused on encoding at the level of tokens and words. To call the model you only have to give the repository address, which in this case is “sentence-transformers/paraphrase-MiniLM-L6-v2”. The numerical vector that is returned to us for each text will be normalized, since the Kmeans model is sensitive to the scales of the inputs. The vectors created have a length of 384. With them what we do is create a dataframe with the same number of columns.
3) Finally we obtain the dataframe from the embedding, which will be the input of our Kmeans model.

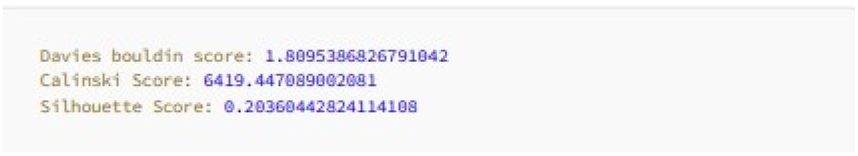
Infact , this procedure has replaced preprocessing.

On the outliers part ,we apply the method already presented to detect outliers, ECOD. We create a dataset that does not contain these types of points.

For the Modeling part again , first we must find out what the optimal number of clusters is. For this we use Elbow Method:

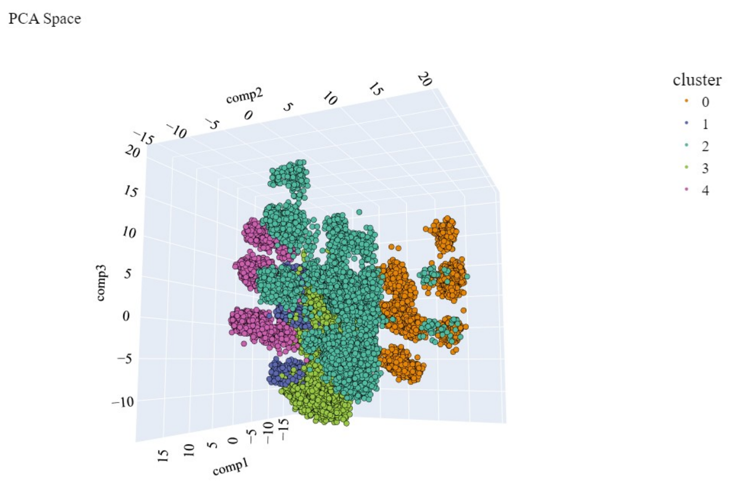


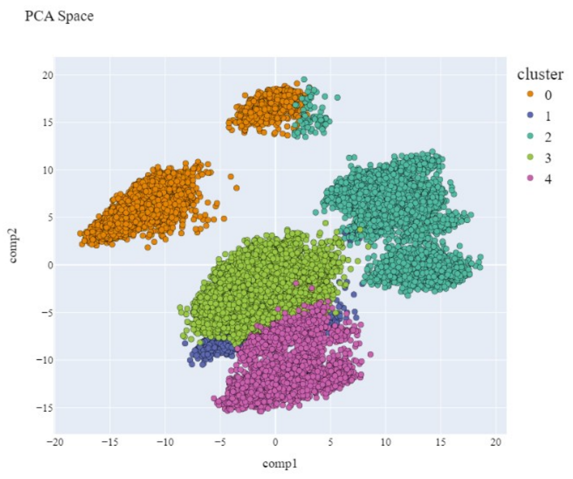
The next thing is to create our Kmeans model with k=5. Next we can obtain the metrics we got on the other methods :



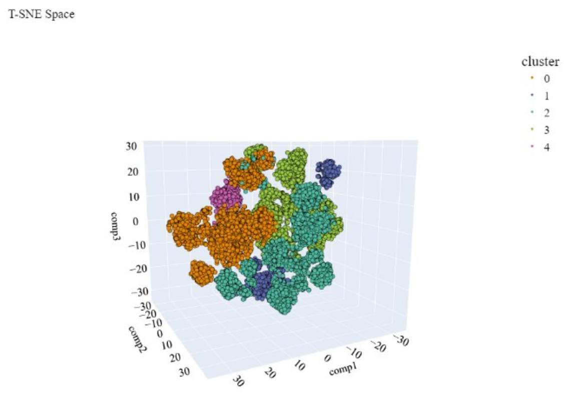
```
Davies bouldin score: 1.8095386826791042
Calinski Score: 6419.447089002081
Silhouette Score: 0.20360442824114108
```

Next , we implement PCA once again and also visualize it’s results. It can be seen that the clusters are much better differentiated than with the traditional method. This is good news. Let us remember that it is important to take into account the variability contained in the first 3 components of our PCA analysis.





Of course we'll implement t-SNE too:



Again, it can be seen that the clusters in the t-SNE are more separated and better differentiated than with the PCA. Furthermore, the difference between the two methods in terms of quality is smaller than when using the traditional Kmeans method.

Conclusion:

Different methods are better at different things and knowing how to implement them all can be quite helpful in machine learning. In most real life cases you have to use multiple of them in a project. Lastly , knowing how to take advantage from the use of LLMs can be be extremely helpful , something we can see in this case because it is clearly seen that the model created with the help of the LLMs stands out.