Machine Learning – Assignment 3

Author: Mohamed Amine DASSOULI

Table of Contents

1. Introduction....2
2. Datasets....2
   2.1. First Dataset: Churn Modeling dataset....2
   2.2. Second Dataset: Car Evaluation dataset....2
3. Clustering algorithms....3
   3.1. K-means....3
   3.2. Expectation Maximization (EM)....4
4. Dimensionality Reduction....5
   4.1. Principal Component Analysis (PCA)....5
   4.2. Independent Component Analysis (ICA)....6
   4.3. Random Projection (RP)....7
   4.4. Random Forest (RF)....7
5. Clustering on dimensionality reduced datasets....8
6. Neural Network on transformed datasets....10
7. Neural Network on datasets with clustering labels....11
8. Conclusion....11

1. Introduction

In this report, we are first going to discuss two clustering algorithms (K-means and Expectation Maximization), and four dimensionality reduction algorithms (Principal Component Analysis, Independent Component Analysis, Random Projection and Random Forest). Then, we are going to do clustering on data with reduced dimensionality. Next, we are going to do classification using a neural network on the dimension reduced data. In the end, we are going to use clustering algorithms to preprocess our data, use the clusters as features and do classification once again with a neural network.

The same datasets from Assignment 1 have been used, more details are going to be provided in the next part.

2. Datasets

2.1. First Dataset: Churn Modeling dataset

It is a binary classification problem about Bank Customer Churn prediction, the goal is to predict whether a customer is going to churn or not.

The dataset is very imbalanced, the positive class has a ratio of *20.37%*. This imbalance makes it very interesting to study. How will the clustering algorithms and the dimensionality reduction techniques handle it ? Will they reduce the domination of the majority class and improve the results of the neural network ?

The initial shape of the dataset is (10000, 13). After removing the target class and the irrelevant features, and encoding the categorical features, our new dataset has 10000 lines and 12 numerical columns.

2.2. Second Dataset: Car Evaluation dataset

It is a multiclass classification problem about evaluating the quality of a car offer based on its physical qualifications, there are four labels: *"unacceptable"*, *"acceptable"*, *"good"* and *"very good"*.

This second dataset is very imbalanced as well, so we are going to merge the labels *"good"* and *"very good"* to reduce the imbalance since it is not the interesting part of the dataset. What makes it interesting is its structure: it is made according to a structural decision model, and the lines represent all the possible combinations of the features. How will clustering and dimensionality reduction techniques handle this type of data ?

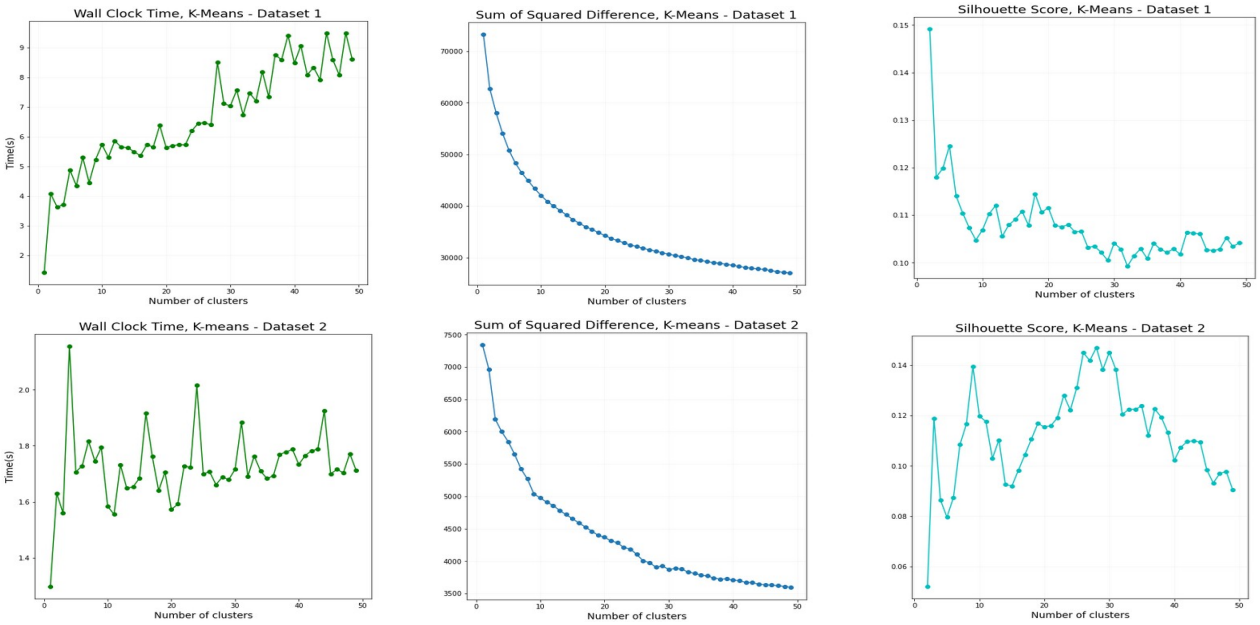
The initial shape of the dataset is (1728, 7). After removing the target class, and encoding the categorical features, our new dataset has 1728 lines and 21 numerical columns.

3. Clustering algorithms

3.1. K-means

The algorithm first picks randomly K centroids, and each point is associated to the cluster whose center is the closest. The centroids are then recomputed by averaging the clusters' points, and the operation is repeated until convergence. We chose to use the Euclidean distance because it performed well for those same datasets with the k-nearest neighbors in the First assignment.

Here are the results for K-means on the two datasets using different K values:



■ The Wall Clock Time is higher for the Churn dataset than the Car dataset because there are more samples (10000 comparing to 1728), and therefore more points to associate to clusters at each step. And overall, the Wall Clock Time gets bigger with higher values of K since there is generally more computation to do to find closest centroids, and for the model to converge (more visible for Dataset 1).

■ Concerning the number of clusters, choosing a very big K makes no sense since each cluster becomes a single point (extreme case), and choosing K=1 leads to a single cluster for all the points, so we need to choose carefully its value. To do so, we can use the elbow method: based on the curve of sum squared error (*SSE*) per number of clusters, we choose K such that adding another cluster does not reduce much the SSE.

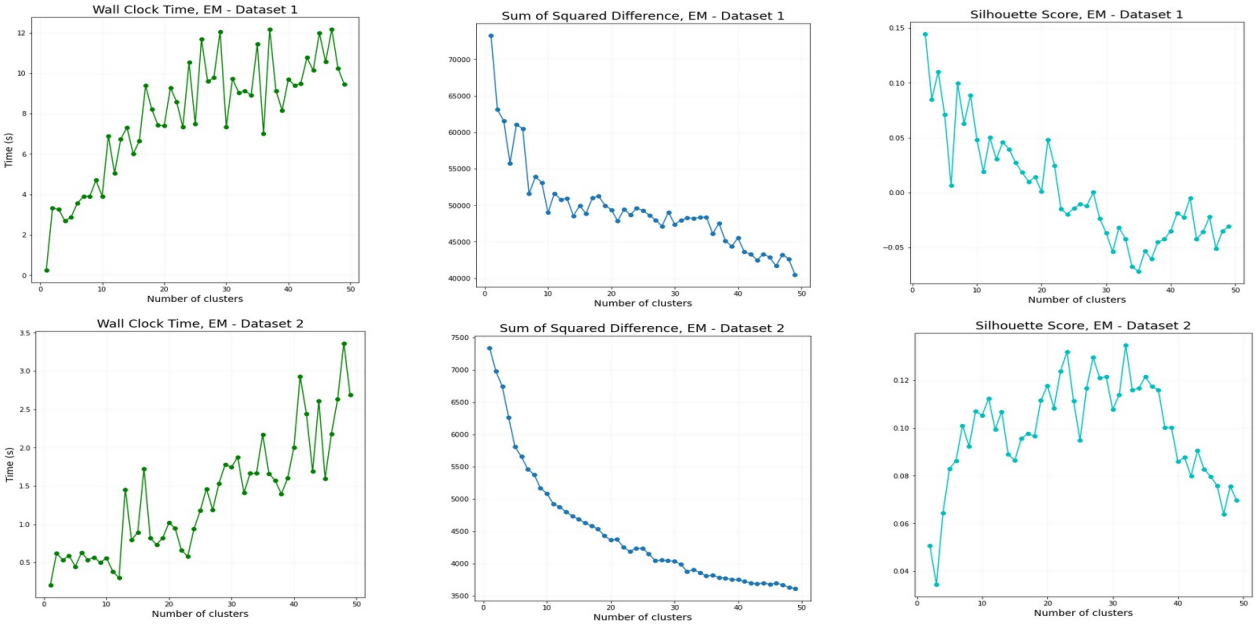
■ In order to choose K, we can use the Silhouette analysis as well. The silhouette score gives a measure of how close each point in a cluster is to the neighboring clusters. A higher value represents a higher separation distance between the resulting clusters.

■ SSE measures intracluster distance, and the Silhouette score gives us an insight on the distance interclusters, so the best way to choose K is to use both of them. For the Churn dataset, using the elbow method, K is between 10 and 30, and silhouette score is maximal for K=18, so it's the best value to choose. For the Car dataset, we can choose K = 28.

3.2. Expectation Maximization (EM)

Contrary to K-means which is a hard clustering method, Expectation Maximization (*EM*) is a soft clustering method, which estimates probabilities of each point belonging to each cluster, and can therefore put a point on different clusters. It starts off by randomly initializing centroids, and computing probabilities. Then, the centers are recomputed based on the probability distribution, and the process is repeated until convergence.

Here are the results for EM on the datasets using different values for number of clusters:

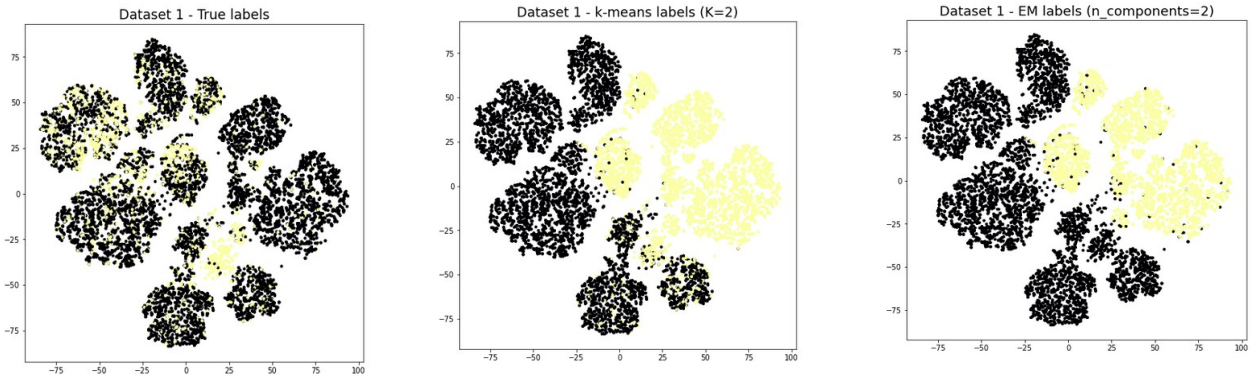


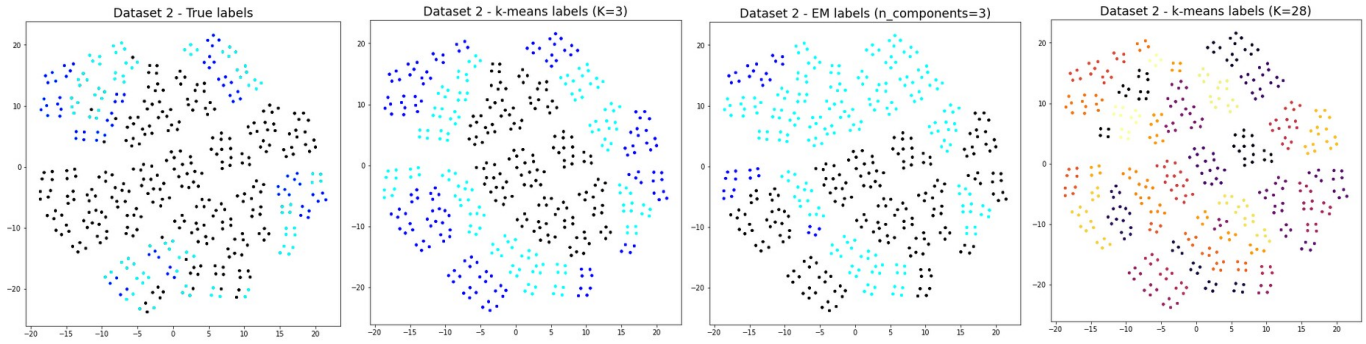
■ Based on the Wall Clock Time figures, EM takes more time than K-means in average.

■ We can once again use the elbow method and silhouette analysis to determine the optimal values of K (or number of clusters). From the graphs, we can choose K=28 for the Churn dataset and K=22 for the Car dataset. These values are different from the ones for K-means because the two algorithms assign points to clusters in a very different way, and what can be best for one is not necessary best for the other one as well.

3.3. Representation of clusters

For visualization, we use t-distributed stochastic neighbor embedding (*t-SNE*), this reduces the high dimensional space of our datasets to 2D. We consider only 2 clusters for dataset 1, and K=3 for dataset 2, to make comparison with the true clusters (labels).





■ The Car Dataset (Dataset 2) has a simple decision boundary compared to the Churn Dataset (Dataset 1), where the clusters are inter-crossing.

■ For the Churn dataset, the clusters of both K-means and EM are not similar at all to the real clusters, because the problematic is difficult (the clusters are mixed), and the two algorithms tend to create distinct separate clusters. Between the two, EM is more suited than K-means because it creates some inter-crossing thanks to its probabilistic nature (we can see some black points in the yellow cluster).

■ Concerning the Car Dataset, the k-means clusters are close to the real clusters. The three clusters represent: *"unacceptable"* car offer (black), *"acceptable"* car offer (cyan), and *"good"* car offer (blue). Putting the limit between a category and another can be done differently (multiple perspectives), so the clusters of k-means make totally sense. As for the EM, the clusters seem less correct because there are cyan points between two parts of the black cluster, however it is still difficult to make a judgment since the data is high dimensional and we are only looking at a 2D representation.

■ The representation of the 28 K-means clusters shows that the points are divided on small groups, so a cluster includes only points which has very common features. By looking at the true labels, it feels like we can reconstruct them by grouping together some small clusters of 28 K-means.

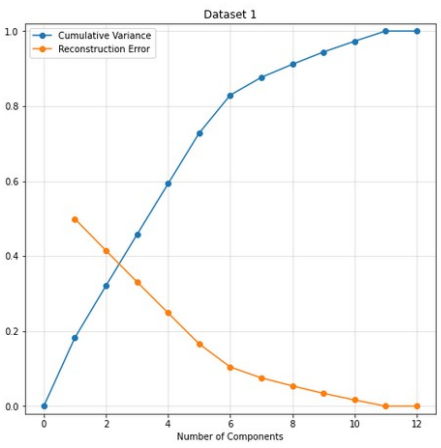
■ **Future improvements:** We can maybe look at higher values of K (the SSE curve of Dataset 1 – EM was not clearly showing a flat zone). We can also change the initialization of EM centroids (for example use K-means result). Also, to choose the best K for EM, we can use other metrics like the Bayesian Information Criterion for example.

4. Dimensionality Reduction

4.1. Principal Component Analysis (PCA)

PCA is a feature transformation algorithm which maps the data to orthogonal axis using linear combinations of the features in order to maximize the variance.

The curves below represent the cumulative variance and the reconstruction error of the PCA components for the two datasets:



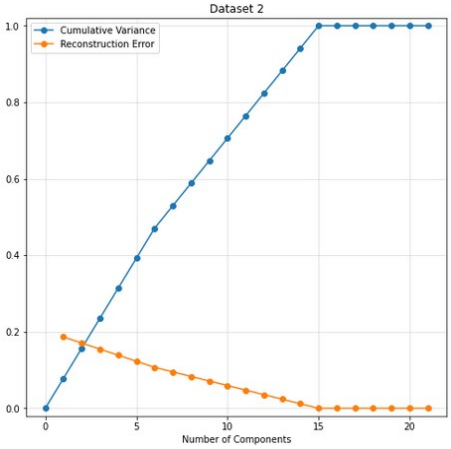


Fig 4.1.a: Cumulative Variance and Reconstruction Error

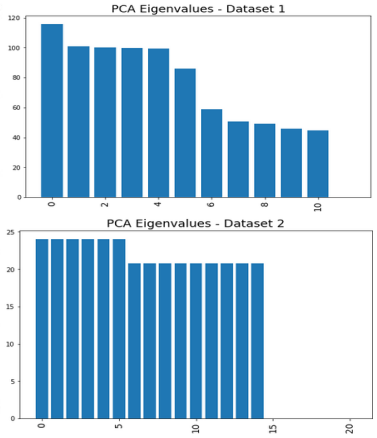


Fig 4.1.b: PCA Eigenvalues

The eigenvalues of the principal components decrease from the first to the last dimension. The first principal components are the most important since they have the highest eigenvalues, the last ones have an eigenvalue almost null, making them irrelevant. For the Dataset 2, the eigenvalues are very close, making the features equally important.

As the principal components capture the variance, we can select only the first dimensions, whose cumulative variance represent 80% for example. This way, we are going to capture most of the information without losing a lot of it (it is reflected by the reconstruction error which becomes very low at that stage). Therefore, we can keep 6 components for the Churn dataset, and 12 components for the Car dataset.

4.2. Independent Component Analysis (ICA)

ICA is a feature transformation algorithm which creates independent components from the features (presumed linear mixtures) by decomposing them and removing correlations.

We use FastICA algorithm for kurtosis maximization. Here are the curves for Kurtosis value and Reconstruction Error per number of components for each dataset:

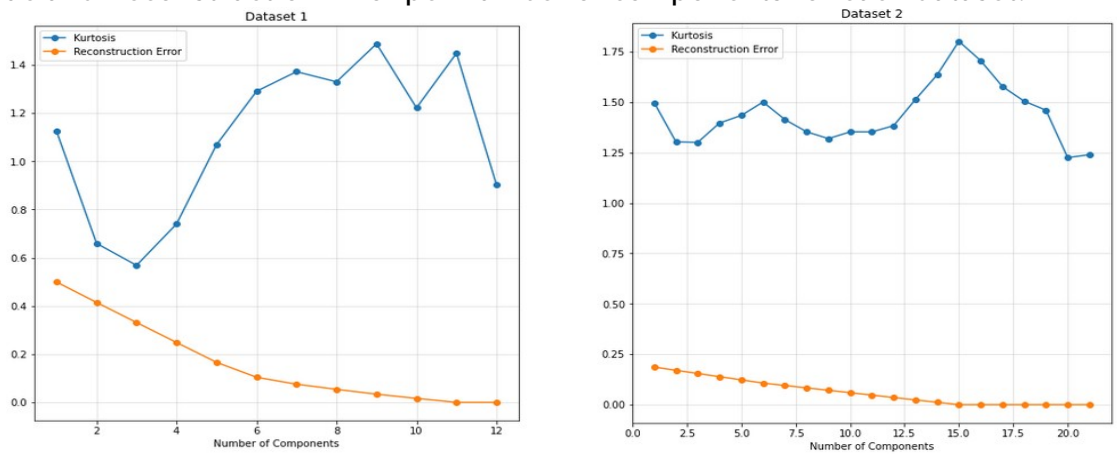


Fig 4.2: Kurtosis and Reconstruction Error per number of components for ICA

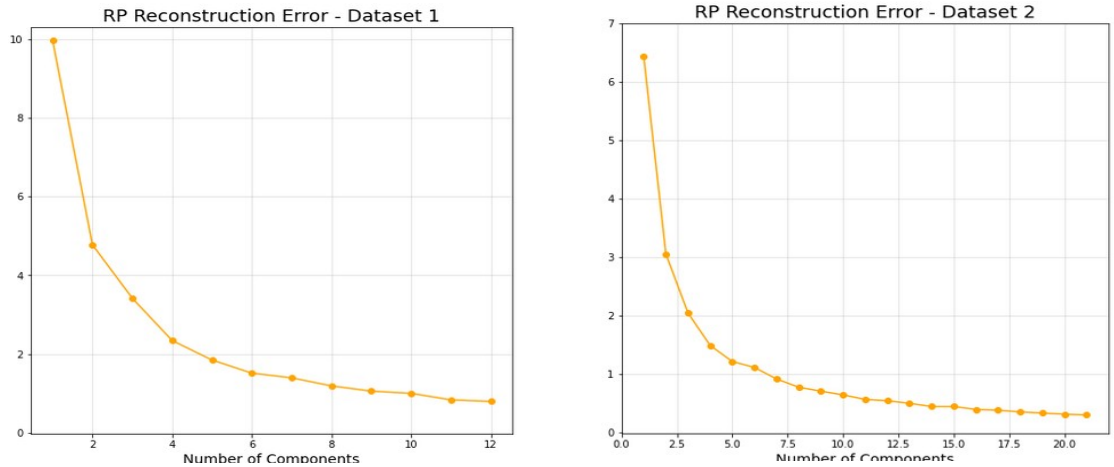
When increasing the number of components, the reconstruction error goes down and reaches 0 at some point, when the independent sources (dimensions of ICA) group all dataset information.

Based on that, the best number of components is 11 for Churn Dataset, and 15 for Car Dataset. Moreover, Kurtosis is maximal for those values which confirms our choice.

4.3. Random Projection (RP)

Random Projection is a very simple feature transformation algorithm, it projects the input space randomly to a lower dimensional subspace.

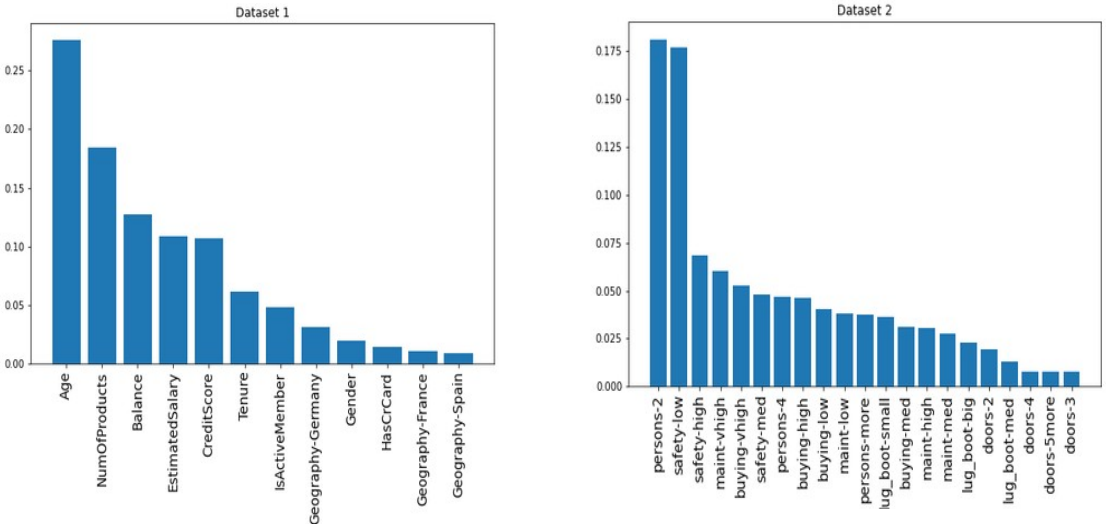
We run RP **100 times**, the curves below show the mean reconstruction error:



RP is fast comparing to PCA! The more components taken, the lower is the reconstruction error, however it doesn't reach 0 because information is lost while projecting on the random dimensions of RP. Based on the elbow method, the best number of components is 5 for Churn Dataset, and around 7 for Car Dataset.

4.4. Random Forest (RF)

Even though Random Forest is mostly used for classification, it can be used to perform feature selection. The algorithm can analyze each feature and assess its importance by looking at its contribution for the target prediction.



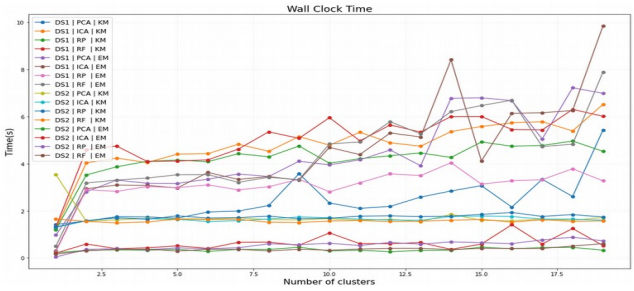
■ For the Churn Dataset, the most important feature is *"Age"* (27.68%), followed by *"NumOfProducts"* (18.47%) and *"Balance"* (12.71%). Several other features are not important like *"Gender"* (2%), *"HasCrCard"* (1.4%), or even*"Geography"* whose three values have an importance less than 3.5%.

■ For the Car Dataset, the two most important features by far are *"persons-2"* (18.11%), which answers the question "Is the car for 2 persons ?" and *"safety-low"* (17.68%), which answers the question "Is the car safety low ?". Overall, without making distinction between values, all features of Car Dataset are important.

We can select features by choosing to keep only the ones that have importance > 4%.

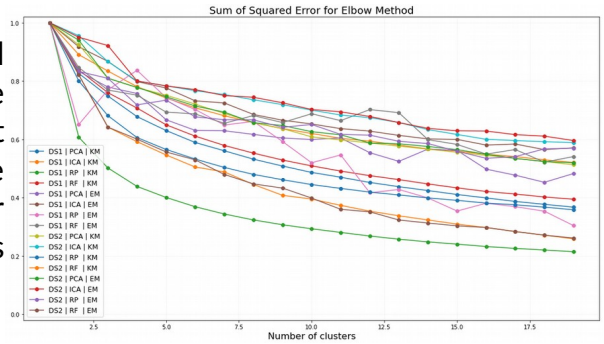
5. Clustering on dimensionality reduced datasets

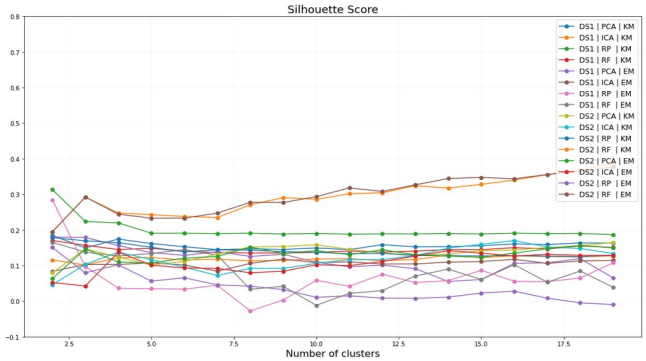
Now, we run K-means and Expectation Maximization (*EM*) on all the reduced datasets.



■ Overall, the clustering algorithms run faster on the dimensionality reduced datasets, comparing to the original ones (in Part 3). It is normal since there are less dimensions and therefore less computation to do.

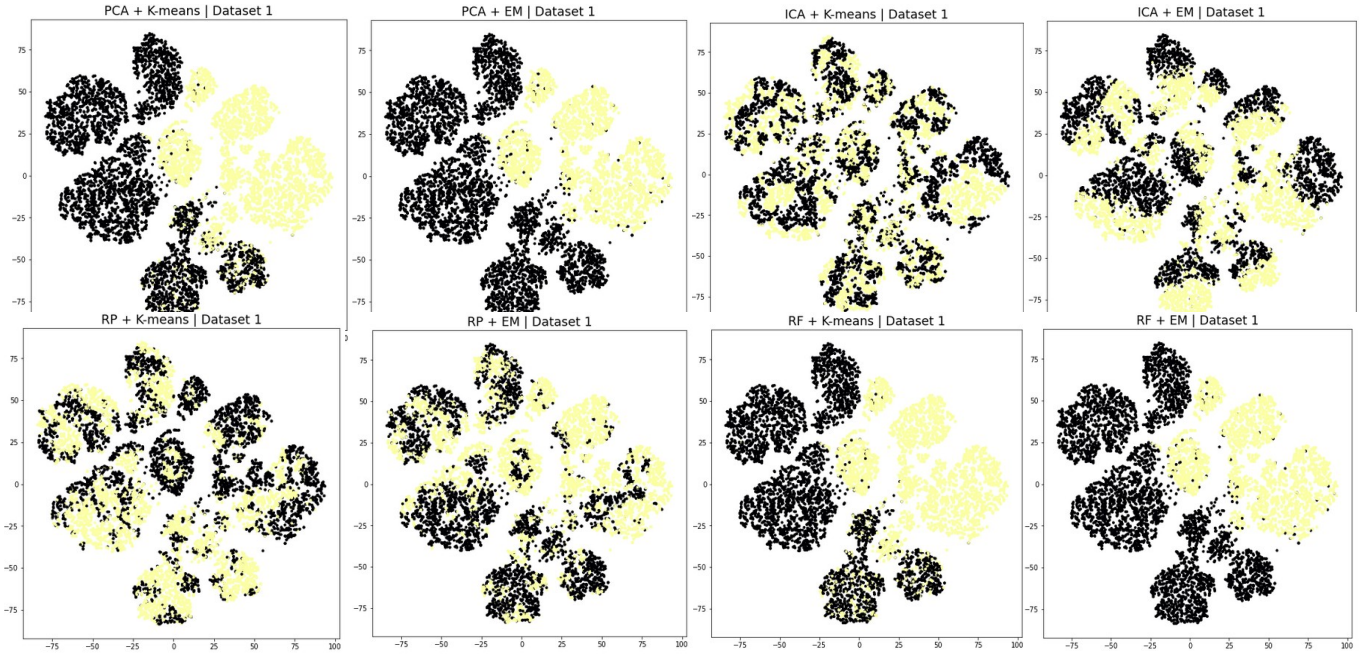
■ In order to represent all Sum of Squared Errors in one figure, we normalized each curve (diving it by its maximum value). By looking at the real values in the notebook, we can notice that the values of SSE have decreased after reducing data dimensions, since the problems became simpler.





■ Similarly, the overall silhouette scores are higher while doing clustering on reduced dimensions. From the plot, the curves with the highest values correspond to clustering after feature selection with Random Forest. It is not surprising because we get rid of irrelevant features, maximizing therefore the distance (separability) between clusters.

For the Churn Dataset, we run K-means and EM considering only 2 clusters in order to compare the resulting clusters with the originals, here is a 2D representation with t-SNE:

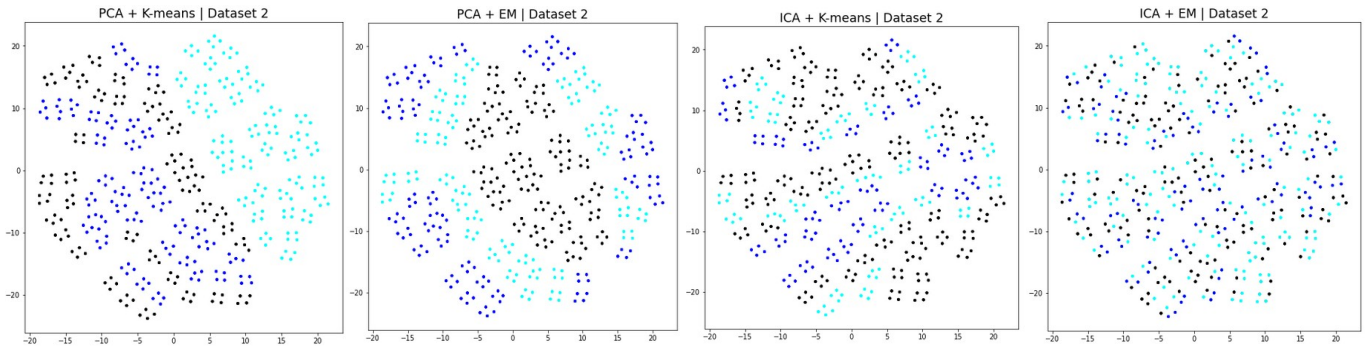


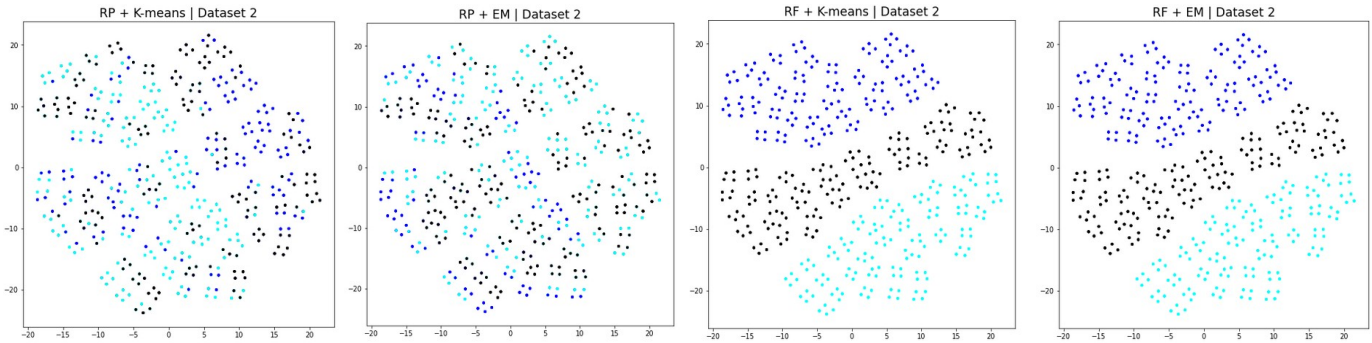
■ The clusters found on PCA and RF datasets are similar to the ones found on the original dataset, they are distinct and visually separable, with just few exceptions on EM. Those points have been attributed to further cluster, with the soft property of EM. This is because those two reducing techniques stick to the original data, PCA's transformation is simply a linear combination of original features, and RF does not transform data.

■ The clusters found using ICA and RP are more inter crossed and mixed than the ones found on the original dataset, they are closer to the true clusters in this sense. It's because ICA create new independent sources as dimensions, and RP's new space is made of random projections, which can make the clustering input noisier. On the other hand, those clusters ignore the detailed structure of the original dataset, since information has been lost during reduction.

■ All the combinations give equal clusters with almost the same number of points, therefore they can't represent the true labels ("*Exited*"=0, "*Exited*"=1), which are imbalanced clusters (20% | 80%). So this imbalance might require to use a higher number of K to get many small clusters representing different categories of clients, and then regroup together the ones corresponding to "0", and the ones corresponding to "1".

Concerning the Car Dataset, we consider 3 clusters, here are the 2D visualizations:





■ Once again, PCA and RF clusters have clear boundaries, while the ICA and RP clusters are inter-crossing and cannot be visually separated (EM ones even more than K-means).

■ Since the dataset is structural and has cohesive clusters (visually separable), PCA and RF, with k-means, are the most suitable for the problematic. ICA and RP lose important information while reducing the space, resulting in noisy clusters, which makes them weak.

6. Neural Network on transformed datasets

In this part, we train an artificial neural network on the dimension reduced datasets (using *PCA*, *ICA*, Random Projection (*RP*), Random Forest (*RF*)), and compare the results on the testing set with the artificial neural network trained on the original dataset. We work on the Churn Dataset rather than the Car Dataset, because most of the features of this second dataset are important, making the dimensionality reduction less interesting.

As a metric, we are going to use the Balanced Accuracy because the data is imbalanced. The architecture of the NN used is (32, 16, 8) for the size of hidden layers, and “relu” as an activation function (more details are provided in the notebook).

For each algorithm, we chose the optimal number of components discussed in Part 4 and **we run 20 experiments**, here are the average results found:

	Original	**PCA**	**ICA**	**RP**	**RF**
Training time	4.77 s	3.29 s	2.89 s	3.30 s	3.87 s
Balanced Accuracy on test set	71.02 %	68.78 %	68.82 %	62.52 %	69.88 %
F1-score on test set	0.56	0.52	0.52	0.4	0.54

■ The training time for NN without any dimension reduction is longer, however the prediction model performs better in term of Balanced Accuracy, probably because there is some information lost in the reduction process.

■ PCA and ICA works quite well, however RP has very low scores in Balanced Accuracy and F1-score. This can be explained by the randomness of its components, which in some of the experiments cannot catch the important information and therefore lower the average over all 20 experiments.

■ RF performs the best on the testing set. This is because Random Forest simply selects the important features, and gets rid of the irrelevant ones. As a result, the prediction model is lighter, the training time gets lower (0.9 *s* less in average), and the efficiency remains almost the same as without dimensionality reduction.

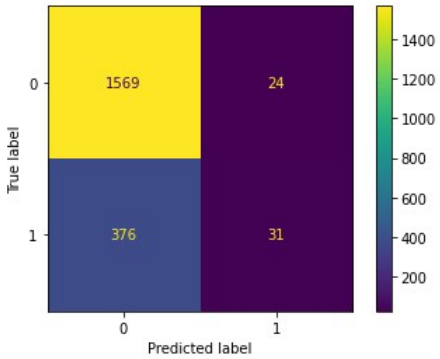
7. Neural Network on datasets with clustering labels

In this last part, we are going to first run clustering on the Churn Dataset, and use the clusters as a feature to fit a neural network. We use K=18 for K-means and K=28 for Expectation Maximization, which are the optimal values for Churn Dataset from Part 3.

Here are the confusion matrices and the results found:

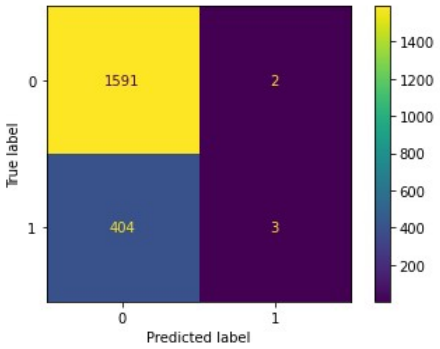
K-means

	precision	recall	f1-score	support
0	0.81	0.98	0.89	1593
1	0.56	0.08	0.13	407
accuracy			0.80	2000
macro avg	0.69	0.53	0.51	2000
weighted avg	0.76	0.80	0.73	2000



Expectation Maximization

	precision	recall	f1-score	support
0	0.80	1.00	0.89	1593
1	0.60	0.01	0.01	407
accuracy			0.80	2000
macro avg	0.70	0.50	0.45	2000
weighted avg	0.76	0.80	0.71	2000



Both Neural Network are very weak, the balanced accuracy is 0.53 for the NN fitted with K-means, and 0.505 for the NN fitted with EM labels. It's probably because the clusters don't correctly represent the real data, and all the important information has been lost during this reduction process (transformation). In order to improve the results, we can increase K, the number of clusters to consider, so that more information is preserved (small clusters are more restrictive and require the samples to be in specific range of data values). We can also change the neural network architecture (add more units or hidden layers) to catch data structure, because it is very complicated one.

8. Conclusion

The dimensionality reduction techniques prove to be efficient against "*The Curse of dimensionality*", they can simplify the machine learning models, make them perform faster, and sometimes even better. But some other times, the important information can be lost in the reducing process and the model becomes very weak.