DIMENSIONALITY REDUCTION AND VISUALIZATION

INTRODUCTION TO MACHINE LEARNING

UNIVERSITAT DE BARCELONA

MASTER IN ARTIFICIAL INTELLIGENCE

UNIVERSITAT DE BARCELONA

Members:

Joël Haupt

Hajar Lachheb

Saurabh Nagarkar

Miquel Sugrañes

Professor: Maria Salamó

Course: 2022/2023

Contents

1. Introduction

The aim of this work is to analyze different representation and dimensionality reduction methodologies and techniques, while trying to cluster data using k-Means and agglomerative clustering algorithms. In the document, we discuss the differences between the application of this clustering approaches to high-dimensional data and reduced-dimensional one. Also, we compare the different techniques used to reduce the dimensionality of data.

In section 2, the original data is represented in graphs, previous to any dimensionality reduction or clustering. In section 3, 4 and 5, we define the implementation of our own PCA code in python, moreover, we discuss its performance with the already implemented codes from the *sklearn* python library, and we study which could be the best number of dimensions before performing a dimensionality reduction to our data. Section 6 we analyze the Feature Agglomeration technique, comparing it to the PCA. In sections 7 and 8 we discuss the different clustering approaches, and we show the results of their application in different data (in terms of dimensionality). Section 9 is a comparison of the PCA algorithm and the t-SNE from *sklearn*. In this section, we compare the graphical results obtained after applying both methods.

As in the previous work, the data sets used for this analysis are the Adult, Pen-based and Vowel data sets, and the preprocessing steps for this data are the same as the ones performed before.

2. Original data sets representation

To visually represent the data sets used in the work, which contain high dimensional data difficult to represent, we performed some tests and combinations to chose two features from each data set to be able to plot them in a 2-dimensional space. The following figure shows the different plots for each data set.

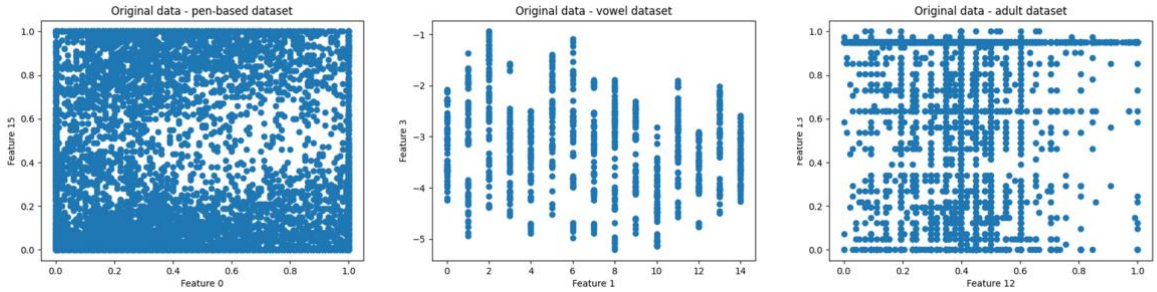


Figure 1 - Original data sets represented in a 2-dimensional space

We can see from the different graphs above (figure 1) that it is difficult to represent the original data, which is defined in a high dimensional space, in an understandable way. The visualization of the data doesn't give us much information about how the data is related or how it is defined in this high dimensional space, apart from knowing the ranges from both features selected and how the instances are defined with them. Moreover, it is difficult to visually find or deduce the clusters of each data set, even knowing in advance the number of classes they have.

3. PCA algorithm implementation

To implement our PCA code we followed several steps that define this algorithm. These steps are summarized in the following:

1. Obtain only the values from the data set (remove columns labels and classes).
2. Compute the mean of each feature and subtract it from each value of each instance.

3. Compute the covariance matrix (we used the *numpy.cov* function)
4. Obtain the eigenvectors and their respective eigenvalues and sort the results by eigenvalues in descending order (we used the *numpy.linalg.eigh* function).
5. Select the first *k* eigenvectors with the highest eigenvalues to create the new low-dimensional space (principal components).
6. Reconstruct the original data set.

After step 5, we already obtain the principal components of the data set (and so the dimensionality-reduced data or transformed data), but we added one more step to reconstruct the original data from the low-dimensional one.

During the execution of the algorithm, the relevant results are shown on console, such as the values of the eigenvectors, the eigenvalues or the covariance matrix, and some graphs show how the data can be visualized in the original and the low-dimensional space.

The plots from data with a high number of dimensions (greater than 3) are created by selecting two features of the data set. If the number of dimensions of the reduced dimensional space is 3 then a 3D plot is created.

4. PCA Results. Comparison to sklearn's algorithms

To compare the performance of our own PCA to the rest of different algorithms used for the dimensionality reduction of our data we decided to use a number of features of *k=2* in order to plot the results in a 2-dimensional space. Moreover, we have chosen the vowel data set as the testing set for the algorithms, since visually is the one that is more understandable and the best to compare the several plots shown in this section.

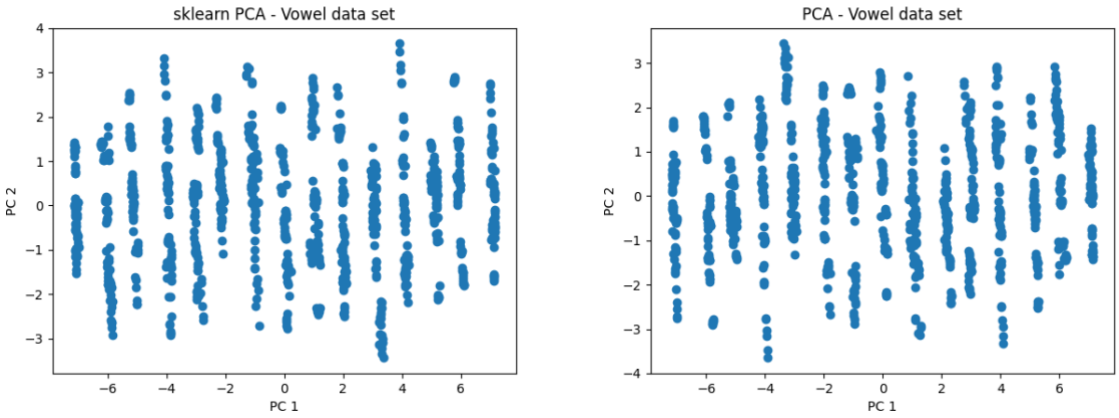


Figure 2 - Reduced Vowel data set result of *sklearn's* PCA and own PCA

From the figure above (figure 2) we can see that both algorithms perform a dimensionality reduction to the input data and transform it to a new low-dimensional space. But analyzing the results obtained we can observe that the plots are not exactly the same. If we look at the first column of points from the first plot, we can see that is the exact same column of points as the last one from the second plot, but upside-down. The same happens for all the rest of points, so turning around 180° (on the paper plane) one plot brings us the other one, and vice versa (somehow like centric symmetry).

We can say then, that our own PCA code returns the mirror of the *sklearn's* PCA algorithm. This is because they don't use the same approach to perform the dimensionality reduction. In our case, we create a covariance matrix and compute the eigenvalues and eigenvectors from that matrix, and the *sklearn's* case is based in in the Singular Value Decomposition (SVD) algorithm, which decomposes the matrix of data and in the process, transposes the principal components. These different approaches are responsible of the differences in the results.

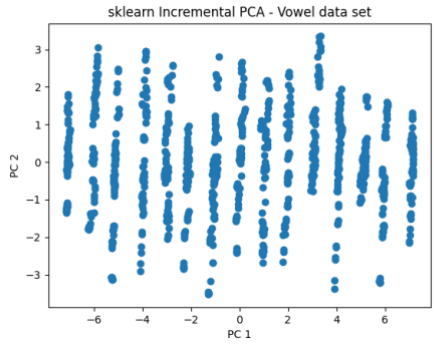


Figure 3 - Reduced Vowel data set result of Incremental PCA

Finally, comparing the performance of the Incremental PCA algorithm to the PCA one, we can see that the results are similar between both algorithms from the *sklearn's* library, but they show some differences. These differences appear because the Incremental PCA algorithm is thought to be used in very large data sets, and it basically down samples the data by building a low-rank approximation for the input data using an amount of memory which is independent of the number of input data samples. This keeps good performance when comparing to regular PCA and becomes more efficient in terms of memory usage. In our case, since data is not so big, this technique is not necessary, and we can make use of regular PCA without an excessive computational cost.

5. Find best k: number of components (dimensions)

We have seen that PCA can be performed to reduce the dimensionality of our data, but how many features we want to reduce or how many principal components we want to keep at the end for our transformed data is not an arbitrary decision. Choosing too many features will mean not reducing as much as possible, while choosing too few might eliminate some relevant information. This tradeoff between information loss and dimensionality reduction can be studied to choose an optimal value for the number of components to keep.

In our case, this number is chosen studying the % of explained variance (ratio of eigenvalues from the covariance matrix) for each of the principal components resulting of the PCA. By doing so, we can define an optimal number of dimensions choosing the number of components that explain or represent most of the variance of the data set. Concretely, we define the optimal number of components for the dimensionality reduction for each data set by selecting the first number of components than represent more than the 80% of the cumulative variance explained.

In the following figure, we can see this analysis in three graphs, one per data set. Each graph shows the cumulative sum of the percentage of explained variance per each component.

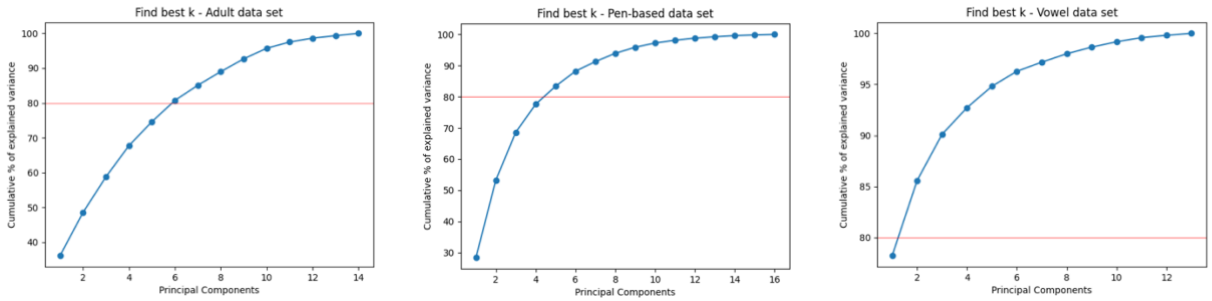


Figure 4 - Explained variance per each component for the different data sets

Analyzing these results, we can see that both Adult and Pen-based data sets need more than 4 components to reach the 80% of explained variance, while the Vowel data set surpasses it already with 2 components. One reason for this might be that the data represented is different, but another reason could also be the number of instances they contain. The first two data sets (Adult and Pen-based) contain more than 40,000 and 10,000 instances respectively, and on the other hand, the Vowel data set contains less than 1,000 instances. After the analysis performed, we obtain the following number of components per data set:

- Adult: 6 components
- Pen-based: 5 components
- Vowel: 2 components

6. PCA results – transformed data sets representation

To compare the results of the original data sets and the resulting transformed data from the PCA, we have decided to reduce the number of dimensions to 2, in order to plot in a two-dimensional space, the most significant components from PCA.

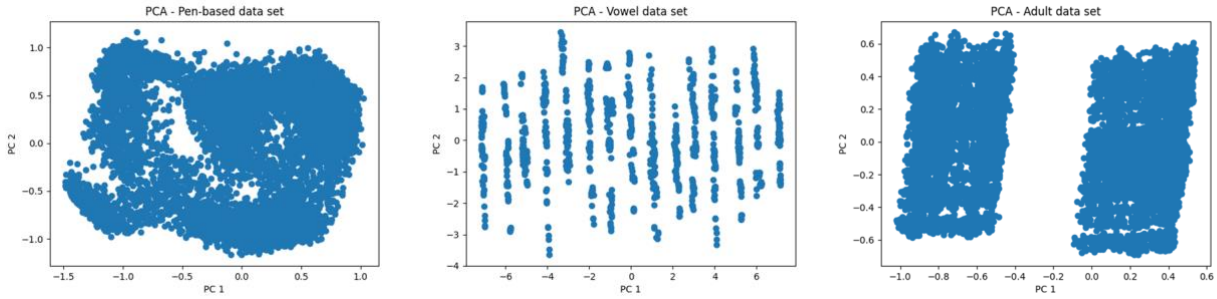


Figure 5 - Resulting data sets from PCA

From figure 5, and referring to figure 1, we can see that the visual representation of the data has changed significantly by applying the principal component analysis to the original data sets. The most relevant changes are shown in the Pen-based and Adult data sets, where both are now more interpretable than they were originally. In the Vowel data set case, small changes in terms of graphical representation can be seen.

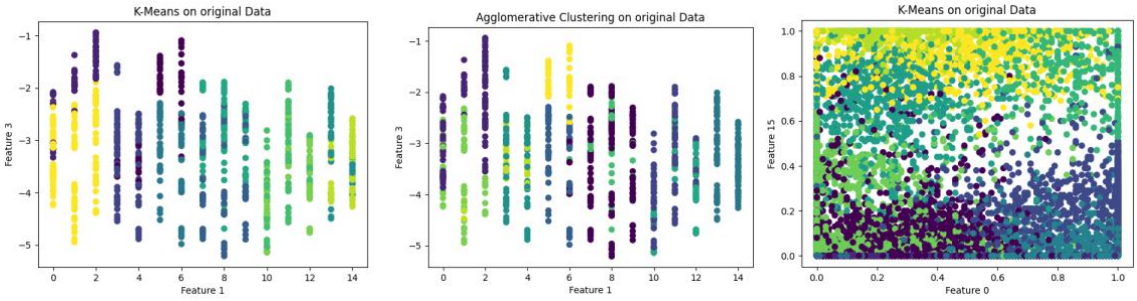
After performing a PCA to the data sets, we obtain more relevant information about how the data is related and represented, and we might infer the clusters for each data set, or at least, we can be closer to find the optimal ones. As we plot the two most relevant components per each data set, most of the variance explained is plotted, while in figure 1 we had to choose two features that we thought better represented the data, without knowing exactly if they were the most relevant in terms of visually defining the data or the relations it has. In any case, the visual information obtained from a graph representation is much higher and clearer after performing PCA.

7. PCA reduced data applied on K-means and Agglomerative Clustering

The previously described PCA was used to reduce the dimensionality of the dataset. To assess the effects of PCA, the original data as well as the reduced data were clustered first with K-means and second with Agglomerative Clustering and were then compared to each other.

The results of the K-means and Agglomerative Clustering are presented and compared to the performance of the clustering on the basic datasets.

On Figure 6 the original data is plotted again with the resulting clusters from each method. The patterns differ among the different cluster algorithms except the adult dataset with only 2 classes have equal clusters.



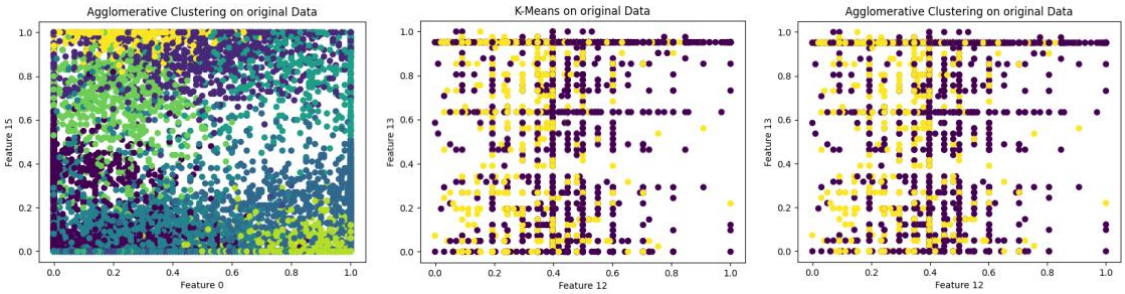


Figure 6 - 2-dimensional plot of clustering on original datasets (in the order Vowel, Pen-based, Adult)

Figure 7 shows a 2-dimensional representation of each dataset with their clusters assigned by either K-means or Agglomerative clustering. The first two dimensions are those components with highest variance, however the dimensionality for each dataset was chosen as shown in Figure 4. In addition, as the k-parameters, the ones from the last project which were determined to be the most suitable for each dataset, were used. I.e., for the vowel dataset k = 11 and the dimensionality is 2, for the pen-based dataset k = 10 and the dimensionality is 5, and finally for the adult dataset k = 2 and the dimensionality is 6.

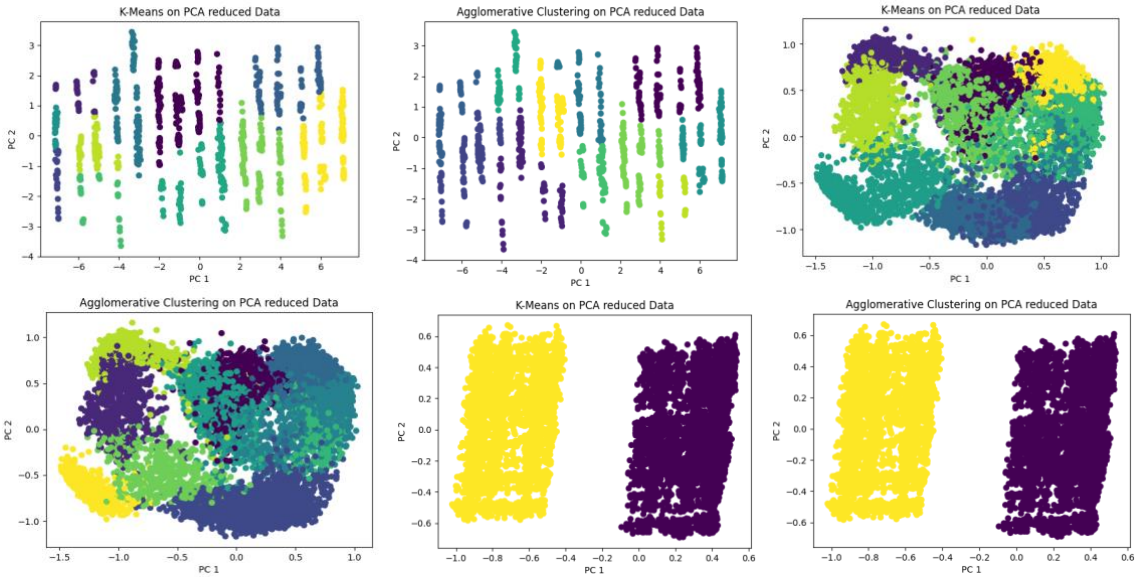


Figure 7 - Clustering on PCA reduced datasets (in the order Vowel, Pen-based, Adult) with the two features with highest variance

	K-means standard data	K-means PCA data	Agglomerative Clustering standard data	Agglomerative Clustering PCA data
Accuracy	0.1020	0.0889	0.0788	**0.1051**
F-score	**0.0992**	0.0827	0.0695	0.0989

Table 1 - Performance scores on vowel dataset

Table 1 shows the performance of each clustering algorithm on either the standard or PCA-reduced vowel dataset. The K-means algorithm has slightly worse performance on the PCA-reduced data compared to the standard dataset. Agglomerative Clustering seems to improve when PCA is applied to the data, both accuracy and the f-score are higher.

	K-means standard data	K-means PCA data	Agglomerative Clustering standard data	Agglomerative Clustering PCA data
Accuracy	0.1245	**0.1799**	0.0002	0.1056
F-score	0.1276	**0.1929**	0.0002	0.0682

Table 2 - Performance scores on pen-based dataset

Table 2 shows the performance of each clustering algorithm on either the standard or PCA-reduced pen-based dataset. According to the metrics K-means performs almost twice as good when applied to the PCA-reduced data than without reduction. The results for the Agglomerative Clustering also greatly improve for the reduced dataset (accuracy around 500 times higher).

	K-means standard data	K-means PCA data	Agglomerative Clustering standard data	Agglomerative Clustering PCA data
Accuracy	0.4965	**0.5035**	**0.5035**	**0.5035**
F-score	0.5171	**0.5282**	**0.5282**	**0.5282**

Table 3 - Performance scores on adult dataset

Table 3 shows the performance of each clustering algorithm on either the standard or PCA-reduced adult dataset. Slight improvements of the K-means applied to the PCA-reduced dataset over the standard dataset are recorded. For Agglomerative Clustering the data reduction does not change anything.

To analyze and conclude, the above obtained observations have to be considered according to their corresponding dataset. Since the adult dataset was too large for the computational expensive agglomerative clustering, which resulted in memory errors, only the first 60% of the dataset were used.

In the vowel dataset the reduction only leads to slight changes in the results. A possible reason for this could be that most data is already contained in the first 2 components which leads to little loss of information and a more interpretable clear clustering. In the pen-based dataset the data reduction resulted in a better performance of classifying, especially for Agglomerative Clustering, which might be because the data is very complex and spread out, hence the reduction improves clustering. Finally, the adult almost had the same accuracy and f-score values. This could be because the dataset is very clearly separated and has already high performance scores.

The average runtime for each dataset for the original data as well as the reduced data (including both PCA and FA) was recorded. The average runtimes were 2.47s original and 0.85s reduced for the vowel dataset, 26.0s original and 21.73s reduced for the pen-based dataset and 35.37s original and 28.31s reduced for the adult dataset. As a result, the reduction techniques improved run time in all cases.

Given the results and due to the advantage of reducing computational costs, PCA proves to be a favorable improvement to the clustering. Although information goes missing, PCA is able to use the important parts of the data to improve the overall results.

8. Reducing data with Feature Agglomeration applied on K-means and Agglomerative Clustering

In this section, instead of PCA another dimension reducing technique was applied called feature agglomeration (FA). Feature Agglomeration is a dimensionality reduction tool that basically uses agglomerative clustering to group together features that look very similar, thus decreasing the number of features. For our tests, we defined the same number of features to reduce to for the FA algorithm as the ones obtained in the analysis for the best k value for the PCA method.

Figure 8 shows each dataset reduced by means of feature agglomeration and clustered according to K-means or Agglomerative clustering.

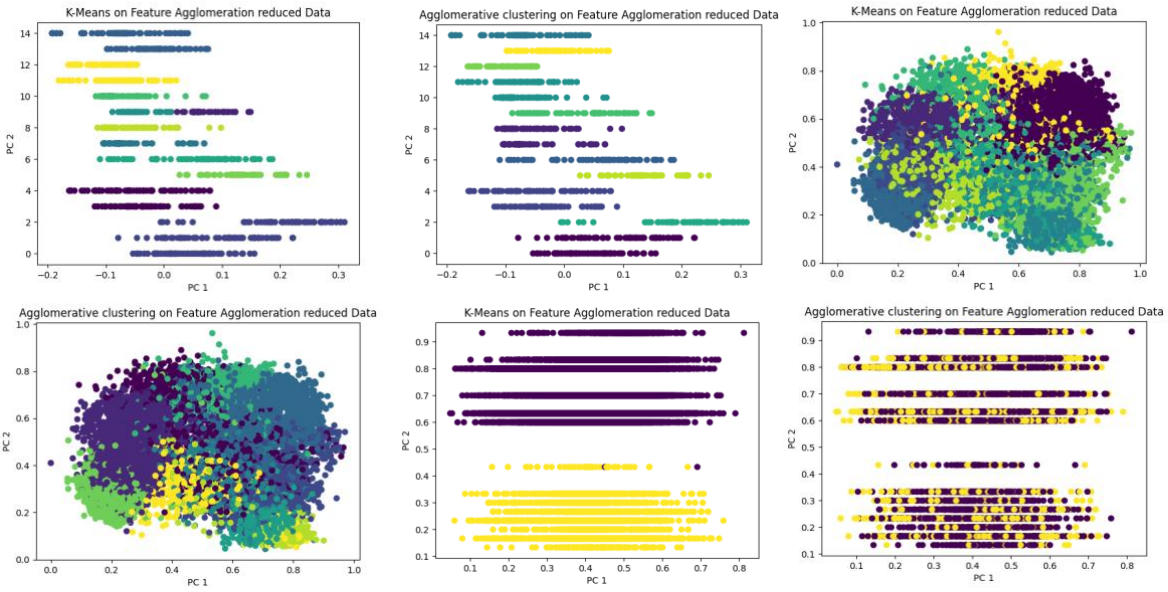


Figure 8 - Clustering on FA reduced datasets (Vowel, Pen-based, Adult) with the two feature with highest variance

	K-means standard data	K-means Feature Agglomeration data	Agglomerative Clustering standard data	Agglomerative Clustering Feature Agglomeration data
Accuracy	**0.1020**	0.0848	0.0788	0.0909
F-score	**0.0992**	0.0758	0.0695	0.0883

Table 4 - Performance scores on vowel dataset

Table 4 shows the performance of the two clustering algorithms applied to the standard and to the Feature Agglomeration reduced vowel dataset. K-means has slightly better performance on the standard data compared to the reduced data. Agglomerative Clustering, however, shows better results on the reduced data.

	K-means standard data	K-means Feature Agglomeration data	Agglomerative Clustering standard data	Agglomerative Clustering Feature Agglomeration data
Accuracy	0.1245	**0.1586**	0.0002	0.1360
F-score	0.1276	**0.1628**	0.0002	0.0991

Table 5 - Performance scores on pen-based dataset

Table 5 shows the performance of the two clustering algorithms applied to the standard and to the Feature Agglomeration reduced pen-based dataset. While K-means slightly improves on the FA-data, Agglomerative Clustering, again, improves by factors of around 500.

	K-means standard data	K-means Feature Agglomeration data	Agglomerative Clustering standard data	Agglomerative Clustering Feature Agglomeration data
Accuracy	0.4965	**0.6479**	0.5035	0.5035
F-score	0.5171	**0.6079**	0.5282	0.5282

Table 6 - Performance scores on adult dataset

Table 6 shows the performance of the two clustering algorithms applied to the standard and to the Feature Agglomeration reduced adult dataset. K-means substantially improves on both metrics and receives the highest overall score among all test runs. As was the case in Section 7, Agglomerative clustering has the same scores for both types of datasets.

To summarize and discuss the results, the performance on each dataset is assessed. The changes due to the feature reduction are similar to the observations in section 7 on the PCA data. It turns out that FA, compared to PCA, gets slightly worse results for the vowel and pen-based dataset, however, it performs better on the adult dataset and receives the highest overall classification score for the K-means in combination with the FA-reduced data. This indicates that due to the different data structures one or the other method can lead to better or worse results.

Generally, given the result of this study, if there are many classes PCA seems to be the choice of method to decrease the data, while less classes work better on FA.

9. Visualization in low dimensional space

In this final section we want to compare two ways to reduce the dimensionality of the data: PCA and t-SNE (t-Stochastic Neighbors Embedding). Both of them will help us reduce the original data set into a low-dimensional space, in order to plot it and being able to visually see the data that we have, similarly as done in section 6 from this document, but now using a 3-dimensional space, and the class labels from the supervised original data (to represent the actual classes).

As from it name says, t-SNE algorithm basically tries to embed the nearest neighbors by modeling each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

We also wanted to analyze the behavior of the techniques in terms of execution time, since we realized that the t-SNE algorithm took much longer to run than the PCA one. The different execution times obtained can be shown in the tables in this section.

a. Adult Dataset

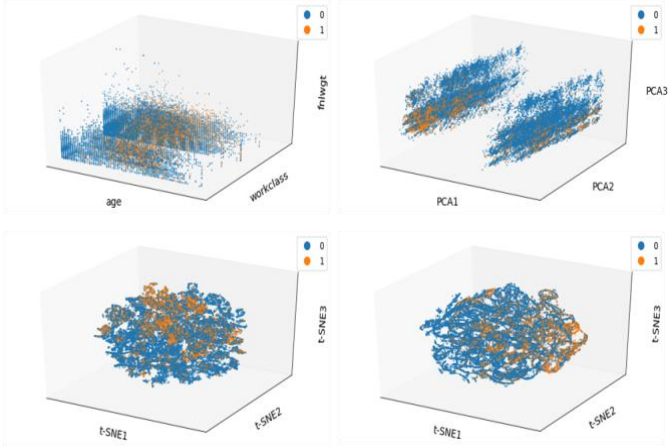


Figure 9 - Plots from Adult data set. Top-left: Original. Top-right: PCA, Bottom-left: t-SNE, Bottom-right: t-SNE with PCA dimensionality reduction

Technique used:	PCA	t-SNE	t-SNE with PCA
Execution time:	**8 Minutes**	80 Minutes	40 Minutes

Table 7 - Execution times for the different visualization techniques - Adult data set

We can see from the results above (Figure 8) that it is difficult to separate the points from each one of the different classes for both PCA and t-SNE and that none of them can clearly show the true clusters and how the data is defined in this 3-dimensional space. We can also see from table 7 that the execution time for the t-SNE algorithm is 10 times higher than the PCA's, which looks inefficient with the results obtained.

b. Vowel Dataset

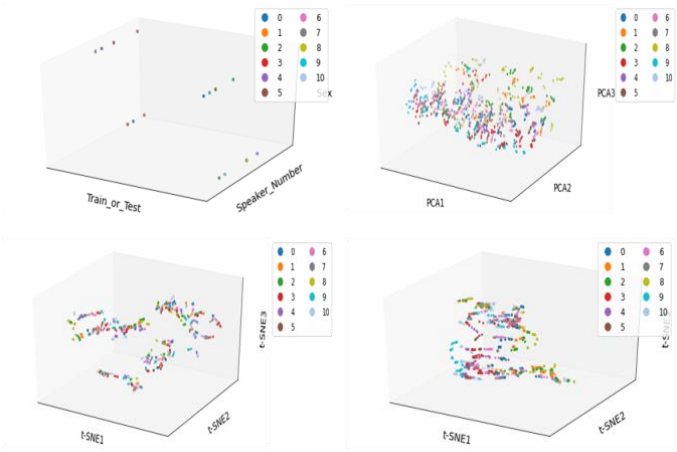


Figure 10 - Plots from Vowel data set. Top-left: Original. Top-right: PCA, Bottom-left: t-SNE, Bottom-right: t-SNE with PCA dimensionality reduction

Technique used:	PCA	t-SNE	t-SNE with PCA
Execution time:	**30 Seconds**	5 Minutes	1 Minute

Table 8 - Execution times for the different visualization techniques - Vowel data set

In this case, it is easy to notice that the short number of instances doesn't help to obtain good results. The vowel data set has few data (less than 1,000 instances), so finding neighbors and relations between features (which are more than 10) is difficult, and so it is the obtainment of good results. We can see this also in the execution times results (table 8), which are really low compared to the ones obtained in table 7 and table 9. We can say from Figure 10 that it is difficult to obtain conclusions from the graphical point of view, since no clear clusters are shown in this 3-dimensional space. Actually, in previous sections we have seen that with 2-dimensions the vowel data set could already be well represented and doing so in a 3-dimensional space seems to add more confusion to the comprehension of the data.

c. Pen Based Dataset

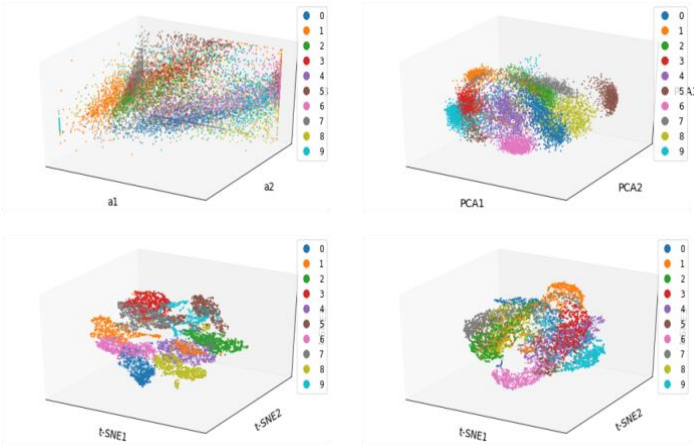


Figure 11 - Plots from Pen-based data set. Top-left: Original. Top-right: PCA, Bottom-left: t-SNE, Bottom-right: t-SNE with PCA dimensionality reduction

Technique used:	PCA	t-SNE	t-SNE with PCA
Execution time:	**3 Minutes**	50 Minutes	10 Minutes

Table 9 - Execution times for the different visualization techniques - Pen-based data set

From this last example we can see that we can obtain better results in terms of visual analysis. From figure 10, it can be observed that for both PCA and t-SNE some proper defined clusters are obtained, and comparing it to the original data, we can define much better our data with the appliance of the algorithms. In this low-dimensional space, groups of similar data are found, so we can say that reducing the number of features does help to interpretate the data in a graphical way. We can also see comparing the PCA and t-SNE result that the second one gives better defined clusters, since it tries to separate the different data from the similar ones. In terms of execution times, we can see from table 9 that even they have increased compared to the Vowel data set, they are still lower than the Adult case. This is because the number of instances in in between of the number of instances from the rest of data sets respectively. Anyway, we can see that running t-SNE takes much longer than running the PCA algorithm.

In any case, we can conclude that the use of PCA and t-SNE, that is, the reduction of dimensions of the original data, improve the visual representation of data, especially when a large number of instances are used. Even it is time consuming, the results and the visual analysis than we can obtain is much better. About the computational cost, in terms of execution time, we can also see that t-SNE is much more expensive than PCA, taking for all data sets much more time to run.

10. Conclusions

During the work several results are shared and shown, the majority of them indicating that reducing the number of features doesn't only help in the visualization of the data sets, but also in the understanding of the underlying information data has. This is important, since machine learning often deals with unknown data, and the relations within the features of it, so understanding and trying to obtain the information that comes from it is a critical task.

We have also seen that, of course, depending on the approach we use for this procedure we might get different results. We would like to emphasize that we don't consider any of them wrong, since they contribute to obtain several points of view of the same data, which actually can look very different. The fact of using different approaches can also help us to determine which one is better suited for the concrete case or task we might have to solve.

In this work, we have analyzed and studied different visualization techniques, in order to reduce the dimensionality of data. We have seen that it is not easy to move from a high dimensional space to a lower one, and to find a good representation of the data. Moreover, it is not the same to plot the information we obtain in a 2- or 3-dimensional space, and this is also interesting and relevant since using one method or the other can better help us to clearly visualize the data we have.

Also, we used the dimensionality reduction techniques to cluster the data obtaining better results. In some cases, as seen in the previous sections, the performance greatly improved by using PCA or FA, and furthermore the runtime decreased significantly .

Finally, the t-SNE-algorithm was applied to reduce the dimension of the data and to plot all datasets (vowel, pen-based, adult) and we found that the execution time was much higher than that of PCA. Also the short number of instances did not help to get better results (in case of vowel datasets). We saw some improvements in the execution time when we used the t-SNE-algorithm with PCA.

Finally, we would like to add that several more approaches could be performed to reduce the dimensionality of data before clustering, as well as to visualize the data, and we could have obtained different results that the ones shown in this document, which could have been compared and maybe even improved the performance of the ones studied.

11. Resources

[1] Scikit learn, hierarchical clustering documentation: https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering

[2] Jalan and P. Kar, Accelerating Extreme Classification via Adaptive Feature Agglomeration, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19):* https://www.ijcai.org/proceedings/2019/0361.pdf

[3] Wikipedia, T-distributed stochastic neighbor embedding: https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

[4] Wattenberg, M., Viégas, F., Johnson, I., 2016. How to Use t-SNE Effectively. Distill 1, e2. https://doi.org/10.23915/distill.00002

[5] Hsu, A., 2020. Visual guide to understanding t-SNE parameters— what they mean. Medium. URL https://medium.com/@ahsu2/visual-guide-to-understanding-t-sne-parameters-what-they-mean-6a8167d61689 (accessed 11.13.22).

[6] Maklin, C., 2022. t-SNE Python Example [WWW Document]. Medium. URL https://towardsdatascience.com/t-sne-python-example-1ded9953f26 (accessed 11.13.22).

[7] Violante, A., 2018. An Introduction to t-SNE with Python Example. Medium. URL https://medium.com/@violante.andre/an-introduction-to-t-sne-with-python-example-47e6ae7dc58f (accessed 11.13.22).