Work 2: Principal Component Analysis

Introduction to Machine Learning

Nikita Belooussov Valeriu Vicol Guillermo Creus

Universitat de Barcelona

Barcelona, March 28, 2022

Contents

1 Description

The purpose of this report is to gain a better understanding of how the principal component analysis (PCA) algorithm and the Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) algorithm work and obtain practical experience with these algorithms. The report will go through the main steps in the PCA algorithm. Then, it will compare the PCA function developed for this report to the already available PCA functions in the *sklearn* [4] library. These will then be compared to the UMAP function found in the *umap-learn* [2] library.

One of the main objectives of this report is to see how reducing a data set affects the performance, visualization, and runtime of a clustering task done by our KMeans implementation (improved with KMeans++ [1]). Our reference method will use the data set with d features, and we will compare it with our own implementation of PCA, *sklearn* PCA, *sklearn* incremental PCA and UMAP. These dimensionality reduction methods will reduce the data set to 2 dimensions so we can plot it and, hopefully, cluster it better.

The data sets that used for this report were the Hypothyroid, Wines, and Cmc. This was due to a requirement of using the same data sets as a previous report. At the end of the report, it is expected that a better understanding of the PCA and UMAP algorithms is achieved. This includes a better understanding of how they work, what are the expected outcomes from them, how they relate to previous reports that were written, and how they compare to each other.

2 Methods

As mentioned in the description section, there are four main algorithms that are going to be tested. The first three are PCA algorithms. The difference being is that one was coded by the group and the other two were implemented from an available Python library called *sklearn*. This library is a commonly used library for machine learning and data analysis. The last algorithm being tested will be UMAP from the *umap-learn* library. This function will not be self developed, due to its complexity. A further explanation of both functions can be found in this section.

2.1 Principal Component Analysis

The largest part of this report was constructing the PCA algorithm. This was done to better understand the individual steps that are done in the PCA algorithm. The steps are:

- Finding the d-dimensional mean vector
- Determining the covariance matrix
- Using the covariance matrix to obtain its eigenvectors and eigenvalues
- Sorting eigenvectors by descending eigenvalues to form a new dimension matrix
- Using the dimension matrix to create new data matrix

The result of several of these steps will be shown and discussed in the results section. The main objective of the PCA algorithm is to reduce the amount of features that are being used. The steps mentioned above help determine which features contain the most information, so that when plotting high dimensional data in lower dimensions, it a good representation of the data is given. This could also be used to lower the amount of features when using machine learning techniques, so that they are able to train faster.

Theoretical basis

In this section, a brief overview of the PCA method will be given, as well as its theoretical basis. First, the data sets used will be denoted with $\boldsymbol{X} \in \mathbb{R}^{d \times n}$. Here, n denotes the number of instances and d the number of features. In addition, from now on $\boldsymbol{X}$ will denote the demeaned data set, which means that every instance (each column) has been redefined as the instance minus the mean ($\boldsymbol{\mu} \in \mathbb{R}^d$) over all of the instances.

With that said, it is time to define some concepts:

- $\boldsymbol{v}$ is said to be an eigenvector of matrix $\boldsymbol{A}$ with eigenvalue λ if $\boldsymbol{Av} = \lambda\boldsymbol{v}$. In addition, a nice property is that $\alpha\boldsymbol{v}$, $\alpha \in \mathbb{R}$, is also an eigenvector of the same eigenvalue if $\boldsymbol{v}$ is already one ($\boldsymbol{A}(\alpha\boldsymbol{v}) = \alpha\boldsymbol{Av} = \alpha\lambda\boldsymbol{v} = \lambda(\alpha\boldsymbol{v})$).
- The covariance matrix of a zero-mean matrix is defined as $\boldsymbol{C} = \frac{\boldsymbol{XX^T}}{n-1}$
- A covariance matrix is symmetric semi-definite matrix, meaning that it will diagonalize. That is, one will be able to find d eigenvectors that will be a base of $\mathbb{R}^d$. Also, this matrix $\boldsymbol{S} \in \mathbb{R}^{d\times d}$ will be orthogonal, i.e., $\boldsymbol{SS^T} = \boldsymbol{S^TS} = \mathbb{I}_d$
- In order to project a demeaned data set to a subspace defined by vectors contained in the matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{d\times n_s}$ one needs to do the following: $\boldsymbol{X}_{\text{projected}} = \boldsymbol{\Lambda^T X} \in \mathbb{R}^{n_s\times n}$. In our case, the subspace will be formed by the set of eigenvectors with the highest eigenvalues: $\{v_1, \ldots, v_{n_s}\}$

Setup

In this report each of the PCA methods are set to return the data back in only two dimensions. It is possible to have the PCA methods to return the data in higher dimensions. The reason for this is that the report is mainly concerned about the visualization of data. This can only be done in two dimensions for the report, since even a three dimensional plot would essentially only be a two dimensional figure on the page. A higher dimension may improve the results of the metrics measured, due to clustering being more correct.

2.2 UMAP

As mentioned before the UMAP part of this report was taken from the preexisting *umap-learn* library. While this function has several inputs, the inputs that will be worked with are the *n neighbor*, *min dist*, and *metric*. The *n neighbor* input controls how the algorithms focuses on the structure of the data. The lower the input value is the more local structures will be shown, while larger input values will start showing the global structures that are present. The *min dist* controls how close points can be near each other. This means that a lower input will produce plots that appear more clustered, while higher inputs will maintain the topological patterns more. Lastly the *metric* is how the distance is measured in this function. For each of the data sets the inputs will be changed. This is to obtain the results that best suit the data.

Setup

In table 1 the setup parameters for this report can be seen. It is important to note that for all of the data sets the euclidean distance was used and that the parameter tuning was done without the color coding, because that would be unfair, since the labels are unknown. These were determined by multiple runs and trying to find setting that provided with clean clustering of the data.

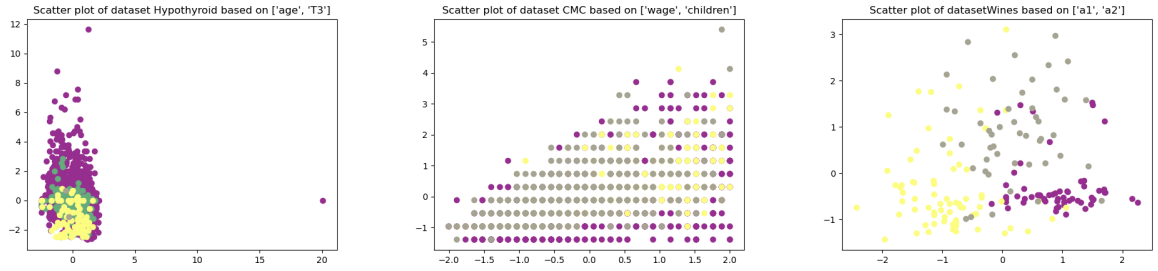
Table 1: UMAP Setup

	Hypothyroid	CMC	Wines
n neighbor	50	50	20
min dist	.05	.01	.05

3 Results

In order to understand the effects of each of these algorithms it is important to see the how the data was displayed before any algorithm was applied. Below one can see an example of this in Figure 1. As seen in the figures, the clustering pattern is not very obvious, except for in the Wines data set, where some clusters can be identified, but instances are intermixed and not well defined. In the other two data sets, the different instances are very intermixed with each other, and the only way of being able to tell how many clusters there are, is with the color coding that was applied. These two data sets provide a perfect example of why these algorithms are required and when they should be used.

In addition, it is extremely important to know the number of classes in the dataset in order to determine if an algorithm has found the correct amount of clusters. For that reason, in table 2 one can see the number of classes



(a) Original Hypothyroid (b) Original cmc (c) Original Wines

Figure 1: Original Data Set

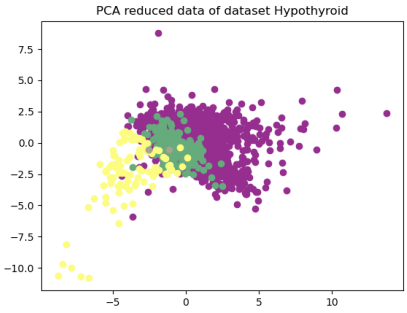
present in each data set, as well as the selected number of clusters in each data set. It is important to note that even though Hypothyroid has 4 classes, it can be considered to have only 3. This is because there are only two instances that are labeled with the fourth label, as a result any clustering algorithm would have a hard time determining these, and would be at the risk of over fitting, if these are classified.

To begin with, the report will discuss the results obtained with the self developed PCA function. The steps can be seen in the description section. It is also important to note that due to how much space the results for each steps take, all of the results can be seen in the Appendix of the report. After obtaining the d-dimensional mean vector, this is the covariance obtained, seen in Table 4 in the Appendix. When comparing this matrix to the ones produced for the hypothyroid data set, seen in Table 10, and the Cmc data set, seen in Table 7, it is seen that this one has a lot more higher values in it. In both of the other data set, a large amount of the matrix values were very low and could be considered negligible. This may be the reason of why it is more difficult to obtain a good representation of the clustering for these data sets, since many of the features are unrelated to each other.

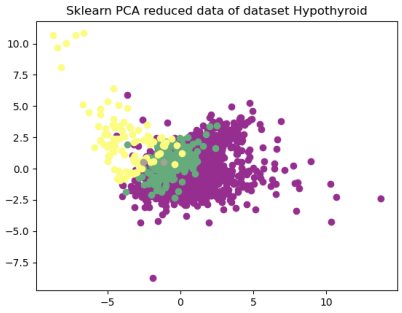
From the covariance matrix, the eigenvalues and vectors can be determined. Again, in the wines data set eigenvectors, seen in Table 6, contain a far less amount of values close to zero than the other two data sets seen in, Table 9 and Table 12. This is an expected result, since it matches the values seen in the covariance matrix. The eigenvalues seen in Tables 5,8, and 11, show this even better. Since for both the cmc and hypothyroid data sets the eigenvalues are extremely close to zero. For each data set there are about two or three high eigenvalues, which could likely be safely removed, without losing information about the data. This information can be represented with the other features. The large amount of low values in both the cmc and hypo thyroid data sets, show that all of the features are very different from each other, and as a result it is unlikely to be able to find two that would be able to show the clustering to the user in a easily understandable way.

The end of this process can be seen in the plots produced in Figures 4a, 3a, and 2a. The PCA clearly helped with the wines data set in making the clustering more separated, but it would still be fairly difficult to separate them visually without the color coding that was added. There does seem to be a lot less intermixing going on than in the original plots. For the hypothyroid scatter plot, it also seems that one of the clusters is more separated out of the general grouping, than it was when compared to the original scatter plot. However, it would still be impossible to tell the different clusters apart without the color mapping. The improvement is that one of the clusters seems to have been slightly separated from larger cluster of data. In the cmc data set, is hard to tell if there was any positive effect on separating out the clusters from each other. The reason for this is likely that there are so many dimensions in the other data sets that have little relation to each other. As a result, finding two or three features that would allow for a good representation would be difficult to find.

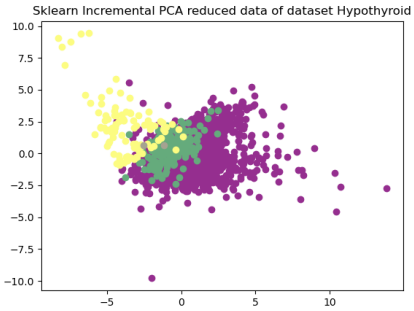
The other part of this report was comparing the different implementations of PCA to see how they performed on the data sets being used. The different output for each algorithm can be seen in Figures 2, 3, and 4. There is not a large visual difference between the different PCA methods tested. This is due to the steps done being largely the same. The main difference is that for *sklearn* PCA function the data is always flipped on its x-axis, when compared to the one that was developed. This is due to the *sklearn* PCA function uses the negative eigenvectors for its calculations, while the one that was developed used the positive ones. This accounts for the flip seen when using the *sklearn*'s PCA function. For the *sklearn* incremental algorithm, it seems like the data is sometimes flipped across the x or



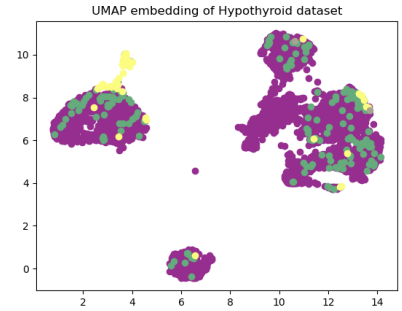
(a) Developed PCA Hypothyroid K-Means



(b) *sklearn* PCA Hypothyroid K-Means

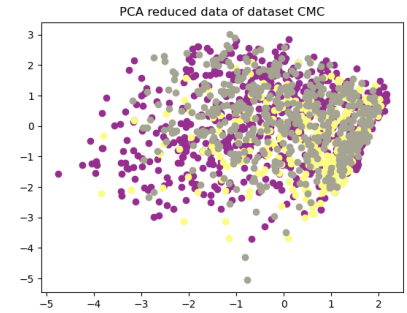


(c) *sklearn* Incremental PCA Hypothyroid K-Means

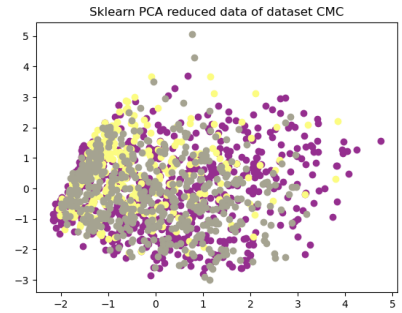


(d) UMAP Hypothyroid K-Means

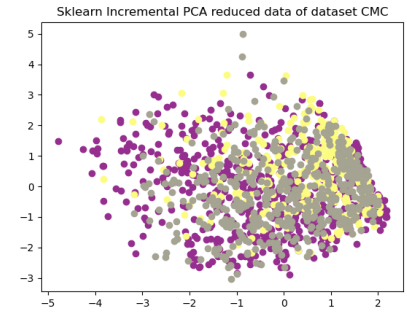
Figure 2: Hypothyroid Algorithm Comparison



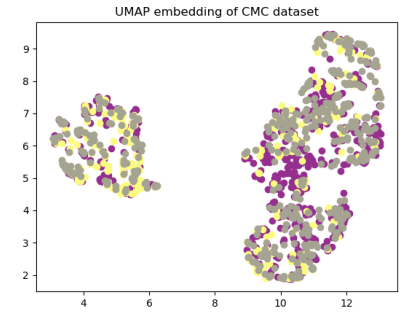
(a) Developed PCA cmc K-Means



(b) *sklearn* PCA cmc K-Means

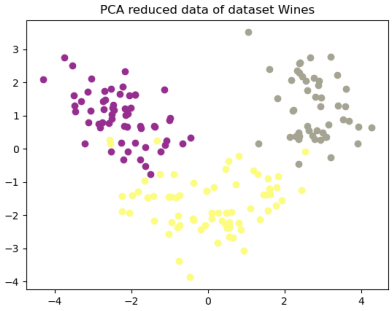


(c) *sklearn* Incremental PCA cmc K-Means

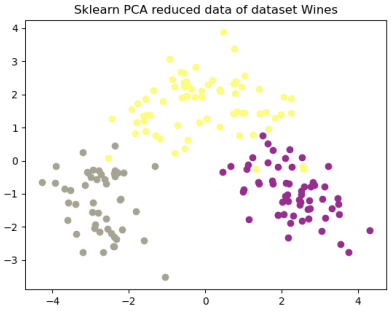


(d) UMAP cmc K-Means

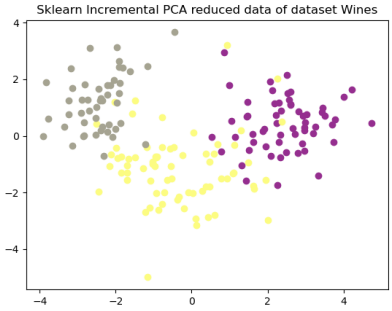
Figure 3: Cmc Algorithm Comparison



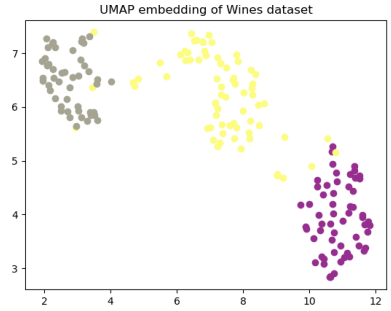
(a) developed PCA Wines K-Means



(b) *sklearn* PCA Wines K-Means



(c) *sklearn* Incremental PCA Wines K-Means



(d) UMAP Wines K-Means

Figure 4: Wines Algorithm Comparison

y axis, depending on the data set being tested, but the overall shape of the data was still the same. Although the incremental PCA is an approximation of the PCA, the reason for the previous observation is the same; eigenvectors can be taken with a positive or negative sign.

Continuing with the previous point, it is important to note that although there have been some reflections on the x-axis and y-axis, this will not change the clustering because it is based on distances, and they are not affected by reflections.

The UMAP algorithm does provide a large difference in the results. It is a lot easier to see clustering in the different data sets. The only data set where it seems to have gotten the clustering correct though is the wine data set. In the wine data set the clustering matches the correct labels set by the data set. There is a little intermixing of the one of the clusters with the other two, but overall it is mostly correct. In the cmc results, Figure 3d, it seems like there are two clusters, possibly three. The correct number of clusters is three, but the main issue is that the clustering seen in cmc is clearly incorrect. All three of the clusters contain intermixing label types within them and not just singular one. A similar thing can be seen with the hypothyroid data set, Figure 2d, where there appears to be 3 main clusters. The correct number of clusters is 4. There are only 2 instances labeled as the 4th cluster in the data set, so it would be hard for any algorithm to determine that there are four. Even with 3 clusters, when looking at the colored values, it is seen that these clusters are not actually clustered by the labels, but by some other relation found by the UMAP algorithm.

Another important result to look at is how the different data transformationa are affecting the internal metrics, that is, how data points are arranged. The metrics being used will be the same that were used in the previous report: Silhouette, Calinski, and Davies Bouldin metrics. These mainly look how well the clusters are created from the data. The results for these metrics can be seen below in Figures 5, 6 and 7.

The internal metrics used in the Hypothyroid data set are shown in figure 5. In the complete data set, developed PCA and *sklearn* PCA we have chosen 3 as the number of clusters because we have a relative maximum (minimum) in the Silhouette (Davies Bouldin) plot. The UMAP plot is pretty straightforward, since at 3 as the number of clusters one

can identify a local maximum in Silhouette and Calinski and a local minimum in the Davies Bouldin score. Lastly, the incremental PCA is difficult to interpret because we cannot find a local maximum in the Silhouette or Calinski method that matches a local minimum in the Davies Bouldin score. However, since our objective is to find a high (low) Silhouette/Calisnki (Davies Bouldin) score, the number of clusters chosen is 2, since it is the best compromise.

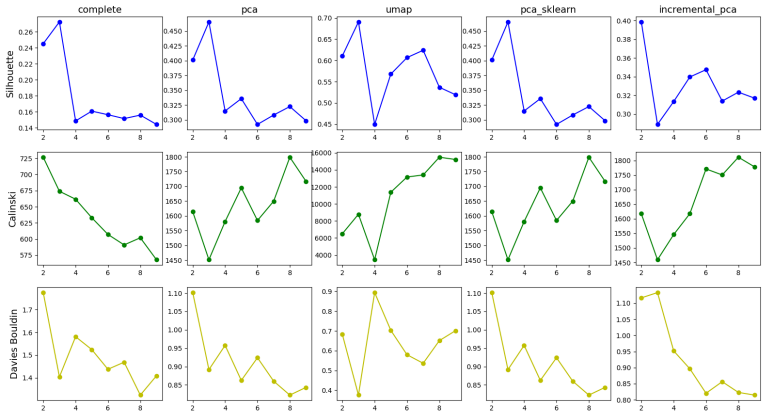


Figure 5: Hypothyroid internal metrics

The internal metrics used in the Cmc data set are shown in figure 5. The complete data set will use a number of clusters equal to 3, since the Davies Bouldin metric presents a minimum at that point. The developed PCA method will also use 4 as the number of clusters since the local maximum in the Calinski metric matches the local minimum in the Davies Bouldin metric. The UMAP is going to use 3 clusters since the Calinski presents a maximum at this point. Also, the *sklearn* PCA will use 4 clusters because this point has a Calinski local maximum in combination with a Davies Bouldin local minimum. Lastly, the incremental PCA will use 4 clusters since there is a local maximum (minimum) in the Calinski (Davies Bouldin) metric.

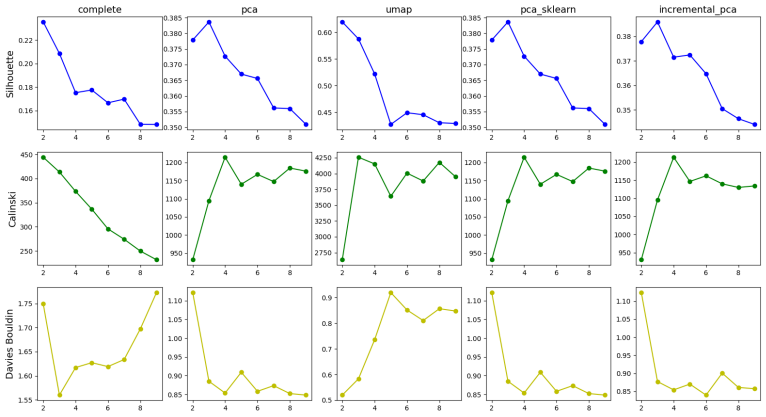


Figure 6: Cmc internal metrics

The internal metrics used in the Wines data set are shown in figure 5. In this data set the number of clusters chosen is 3, since at this point one can observe a local maximum (minimum) in the Silhouette and Calinski (Davies Bouldin) metric.

An important note to take into account when selecting the number of clusters is that these plots were obtained

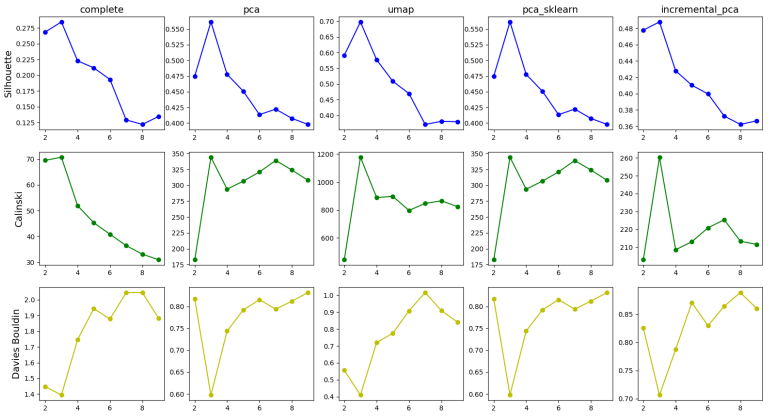


Figure 7: Wines internal metrics

after only one run. That is, a methodology that uses multiple instances was not put in place, so it is possible that the seeding of the initial points could change the metrics' values and thus, how well the algorithm performed. Summarizing, if different initial points were chosen the algorithm would have a better (or worse) performance.

Looking at table 2 one can see how each algorithm selects the number of clusters. It does appear that the different algorithms had a hard time predicting the number of classes in the Hypothyroid data set, and all of the methods predicted less than 4 clusters. One of the reasons could be that one of the classes only has two points of data so it is reasonable not to expect any clustering algorithm to be able to differentiate them. In Cmc most of the methods predicted 4 clusters, when there were only 3, and the Wines data set was the easiest to predict because all of the methods gave a correct prediction. Lastly, it is important to point out that UMAP and the complete data set were the methods that predicted the number of clusters better.

Table 2: Selected number of clusters in the data sets used

	Hypothyroid	CMC	Wines
Number of classes	**4**	**3**	**3**
Complete data set	3	3	3
Developed PCA	3	4	3
sklearn PCA	3	4	3
sklearn Incremental PCA	2	4	3
umap-learn	3	3	3

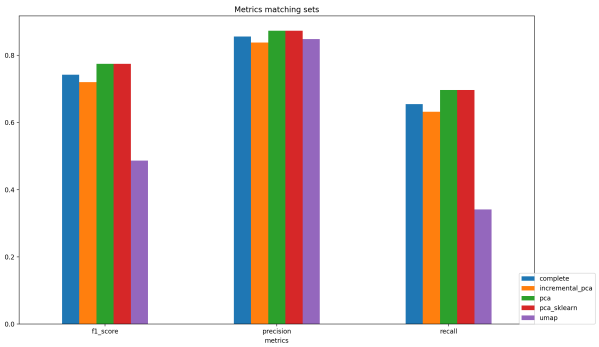
In figure 8 one can see the F1-Score, Precision and Recall for the three data sets, as in [3]. First of all, before analyzing anything, it is important to note that the developed PCA and *sklearn* PCA will give the same results (if the initial points are the same) since they will return the same data. Moreover, the Incremental PCA will yield different results because it builds a low-rank approximation for the input data, so the results will not be exactly the same. Lastly, the metric by default to compare methods will be the F1-Score, which is the harmonic mean of the precision and recall, and thus, it contains information of both metrics.

In the Hypothyroid data set the PCA method with higher number of cluster performs best, surpassing the complete data set. UMAP's performance is sub-optimal, since its recall makes its F1-Score be lowered significantly. Looking at figure 2d it is clear why this happened; the clusters are mixed with instances of different labels.

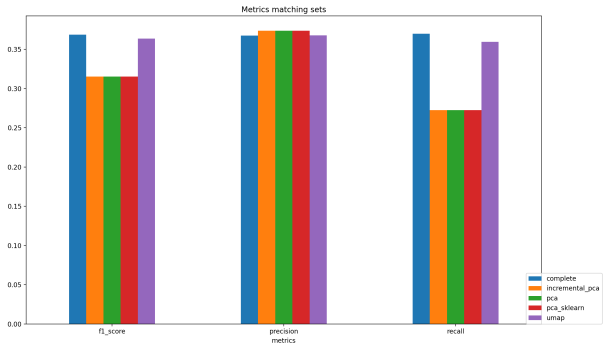
The results of the Cmc data set are very different from the ones observed in the Hypothyroid data set. The three PCA methods chose 4 as the number of clusters, which gave the same results to all of the methods. This goes in line with our past observations about PCA methods: the *sklearn* and developed methods give the same results and the incremental is a good approximation of the past two methods. However, the external metrics of the PCA

methods have not been good, with an F1-Score of 0.30. This is to be expected, because as one can see in figure 3, the PCA reduction only shows one clear cluster, unlike UMAP which obtains external metrics slightly lower than the complete data set. This means that reducing the data set has not brought any improvement.

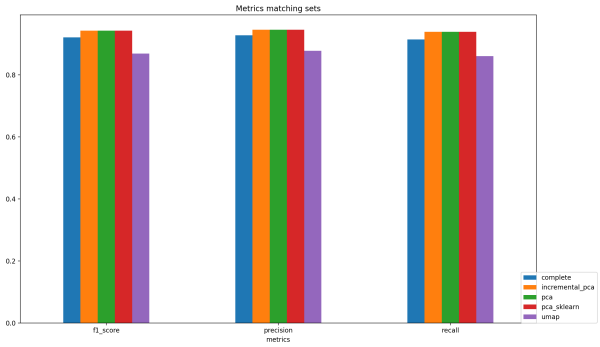
Finally, in the Wines data set one can see that reducing the data set only brings improvement for the PCA methods, because UMAP performs the worst. In this data set it is clear that reducing the data set is not needed because the improvement in the external metrics is negligible.



(a) Hypothyroid data set



(b) Cmc data set



(c) Wines data set

Figure 8: External metrics

The last important effect that these algorithms have that needs to be looked into is how they affect the speed at which the data analysis happens at. In table 3, it is seen how long it takes for each of the algorithms to run using the data set, how long the clustering and labeling takes, and finally the overall time for each of these methods. This is important to understand how helpful these algorithms are with decreasing the runtime due to how many calculations are being done.

There are several outcomes that come out from studying the time results from the table. It is important to note that in this discussion that when a data set is referred to as being smaller, it means having less features and not necessarily having less instances. First, that even though the UMAP increased the speed of clustering and labeling greatly in all cases, the time it spends on transforming the data makes it slower in all of the cases tested. It is likely that if tested with a large enough data set, then this algorithm would be faster than the rest of the ones tested in this report, due to the time it saves in the clustering and labeling part of the computations. The reasoning of why UMAP is so much faster for the clustering and labeling is because the data has been transformed to be more much more clustered than in any of the other methods. As a result, fewer iterations have to be made in K-Means in order for to stop improving.

The different PCA methods mostly followed the expected results. The *sklearn* PCA function was able to run faster

Table 3: Average Timing Results

Algorithm	Algorithm Runtime	Clustering and Labeling	Total
Wines			
Complete Data Table	0	0.0663	0.0663
Self Developed PCA	0.538	0.0773	0.615
Sklearn PCA	0.475	0.0767	0.552
Incremental PCA	0.550	0.0803	0.630
UMAP	8.88	0.0589	8.94
Cmc			
Complete Data Table	0	1.63	1.63
Self Developed PCA	0.679	1.62	2.30
Sklearn PCA	0.534	1.73	2.26
Incremental PCA	0.577	1.53	2.11
UMAP	12.4	1.02	13.5
Hypothyroid			
Complete Data Table	0	6.03	6.03
Self Developed PCA	0.650	4.82	5.47
Sklearn PCA	0.548	4.79	5.34
Incremental PCA	0.780	3.61	4.39
UMAP	16.2	2.32	18.4

than the one that was developed for this report. This is not surprising since we are using *sklearn* functions. The clustering and labeling between these two is practically the same, with the changes in values can possibly be contributed to the computer doing other processes while running the tests, and as a result preforming one slower than the other. The two PCA functions return essentially the same data points, as a result it is expected for them to take the same amount of time during the clustering and labeling parts of the program.

The incremental PCA was clearly slower in the smaller data sets. This is likely due to the time it takes to split the data into batches taking up more time than the amount of time that it saves in computation. This theory is upheld since as the data set becomes bigger the Incremental PCA becomes faster than the rest of the PCA methods. The likely reason of why the incremental PCA method is faster than the rest of the PCA methods for clustering and labeling is that the data points it returns are ordered differently than the normal PCA methods return, this is likely due to the division process for the batches. In this process it is possible that data points are arranged more into clusters, and as a result it takes less amount of iterations to cluster the data, similar to what has happened with UMAP. As a result, it becomes apparent that using these methods depend on how many features the different data sets have, since if there aren't many features then the calculations done for these methods take longer than what the save for the calculations for clustering.

Overall, it appears that these algorithms can be helpful with the visualization of data. While they did not help with the visualization on the scatter plots, they did help in the external metrics. The UMAP algorithm was the most helpful overall. Although it did not cluster the data correctly in two of the data sets, it did provide the correct number of clusters in both instances. There was not a significant difference between the different PCA algorithms used, the only notable difference was that the two algorithms from the sklearn library ended up rotating flipping the data across an axis, depending on the algorithm. This does not really affect the visualization of the data, or the clustering so it isn't very important. Although it useful to keep in mind. As a result, it is likely best to use the UMAP algorithm to help visualize your data, when you have time to adjust the settings, if this is not possible, then using one of the PCA functions is also possible.

4 Conclusion

To conclude this report, the various algorithms used had various advantages when using them to analyze the data sets. The independent steps taken in the PCA helped uncover some underlying information of the data set. This

was mainly how the various features related to each other and how many were fine with removing due to repetitive information. It also provided which features were the most useful to look at, but since there were more than three features for two of the data sets that were contained a lot of information, it was not very useful for this particular case. The amount of features could be reduced in each of the data sets. This can be determined by looking at the eigenvalues, since a few of them have relatively higher number than what they are compared to the rest of the values these factors can be removed.

As mentioned before, there was no real difference between the different PCA algorithms when comparing the one that was developed for the report and the two found in the *sklearn* library. The main difference was that the data was flipped on either the x or y axis, which is completely normal because *sklearn* takes the negative eigenvectors. The external metrics showed that the clustering did appear to function better with the PCA algorithms and UMAP, although it is important to note that even though the external metrics appeared to be better, the scatter plots did not show improvement. There was no apparent clustering in the Cmc and Hypothyroid data sets for the PCA, and for the UMAP the created clusters contained multiple different labels, and not just single one.

Additionally, the clusters observed in the scatter plots of the complete and reduced data sets are very different. The former has unidentifiable clusters and the later starts to separate the clusters, even though most of the instances are mixed. This is clearly depicted in the UMAP's plots, where it can separate data in different clusters but the recall is not very good (it over predicts same-cluster instances).

Overall, the external metrics (F1-Score, Precision and Recall) can only be improved with the PCA reduced data sets. UMAP seems to always trail along and never surpasses the performance observed on the complete data set. Therefore, it can be concluded that PCA reduced data sets provide better data for clustering but one has to be careful when choosing the number of clusters. Since this is a hyper-parameter, we decided to use Silhouette, Calinski and Davies Bouldin, which tend to give good results. However, in the Cmc data set, all of the PCA methods chose 4 clusters when there were 3 true classes. This affected the metrics, and it was the only data set where the PCA methods could not surpass the complete data set's performance.

There was an effect on the overall runtime, which depended on the amount of features the data had has. If the data set has relatively a small number of features, then the fastest method was not using either PCA or UMAP. As the number the features increase, the other methods become more advantages in terms of their runtime. With at first the regular *sklearn* PCA function becoming the fastest, then the incremental PCA function. The report did not get to a point where the UMAP would be the fastest method, but it is very likely that with a enough features, the UMAP method would become the fastest method to use. The difference between the *sklearn* PCA function and the one that was developed for the report mainly came down to the algorithm it self. With the one that was developed being slower as expected. Also as expected the time it took to cluster and label the data sets that these two methods produced also took roughly the same amount of time.

There were not a lot of similarities in the visualization between the PCA and UMAP. The only similarity observed is that if different cluster instances were close (not separated enough), then, even if the UMAP separates the apparent clusters, these instances will still be in the same cluster. This is likely due to the fact that the data sets where this occurs have multiple features with a large variance when compared to the other features. The main difference is that the UMAP did appear to start clustering the data on its own, and was able to get the correct number of clusters.

To end with, this report demonstrates how useful these algorithms are with dealing with large amounts of data. The PCA algorithms helped show which features are not needed, and did help with the external metrics. The UMAP was able to cluster the data into the correct number of clusters and also helped the with the external metrics. The limitation of the number of features will speed up the processing time for clustering, due to less amount of calculations being required. So overall, it is important to understand how and when to use these algorithms to help preprocess data.

5 Appendix

5.1 Wines

Covariance matrices

Table 4: Covariance Matrix

1.01e+00	9.49e-02	2.13e-01	-3.12e-01	2.72e-01	2.91e-01	2.38e-01	-1.57e-01	1.37e-01	5.49e-01	-7.22e-02	7.28e-02	6.47e-01
9.49e-02	1.01e+00	1.65e-01	2.90e-01	-5.49e-02	-3.37e-01	-4.13e-01	2.95e-01	-2.22e-01	2.50e-01	-5.64e-01	-3.71e-01	-1.93e-01
2.13e-01	1.65e-01	1.01e+00	4.46e-01	2.88e-01	1.30e-01	1.16e-01	1.87e-01	9.71e-03	2.60e-01	-7.51e-02	3.93e-03	2.25e-01
-3.12e-01	2.90e-01	4.46e-01	1.01e+00	-8.38e-02	-3.23e-01	-3.53e-01	3.64e-01	-1.98e-01	1.88e-02	-2.76e-01	-2.78e-01	-4.43e-01
2.72e-01	-5.49e-02	2.88e-01	-8.38e-02	1.01e+00	2.16e-01	1.97e-01	-2.58e-01	2.38e-01	2.01e-01	5.57e-02	6.64e-02	3.96e-01
2.91e-01	-3.37e-01	1.30e-01	-3.23e-01	2.16e-01	1.01e+00	8.69e-01	-4.52e-01	6.16e-01	-5.54e-02	4.36e-01	7.04e-01	5.01e-01
2.38e-01	-4.13e-01	1.16e-01	-3.53e-01	1.97e-01	8.69e-01	1.01e+00	-5.41e-01	6.56e-01	-1.73e-01	5.47e-01	7.92e-01	4.97e-01
-1.57e-01	2.95e-01	1.87e-01	3.64e-01	-2.58e-01	-4.52e-01	-5.41e-01	1.01e+00	-3.68e-01	1.40e-01	-2.64e-01	-5.06e-01	-3.13e-01
1.37e-01	-2.22e-01	9.71e-03	-1.98e-01	2.38e-01	6.16e-01	6.56e-01	-3.68e-01	1.01e+00	-2.54e-02	2.97e-01	5.22e-01	3.32e-01
5.49e-01	2.50e-01	2.60e-01	1.88e-02	2.01e-01	-5.54e-02	-1.73e-01	1.40e-01	-2.54e-02	1.01e+00	-5.25e-01	-4.31e-01	3.18e-01
-7.22e-02	-5.64e-01	-7.51e-02	-2.76e-01	5.57e-02	4.36e-01	5.47e-01	-2.64e-01	2.97e-01	-5.25e-01	1.01e+00	5.69e-01	2.38e-01
7.28e-02	-3.71e-01	3.93e-03	-2.78e-01	6.64e-02	7.04e-01	7.92e-01	-5.06e-01	5.22e-01	-4.31e-01	5.69e-01	1.01e+00	3.15e-01
6.47e-01	-1.93e-01	2.25e-01	-4.43e-01	3.96e-01	5.01e-01	4.97e-01	-3.13e-01	3.32e-01	3.18e-01	2.38e-01	3.15e-01	1.01e+00

Eigen Values

Table 5: Wine Data Eigenvalues

[0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.6, 0.6, 0.9, 0.9, 1.5, 2.5, 4.7]

Eigenvectors

Table 6: Wine Data Eigenvectors

$\boldsymbol{v_1}$:	1.50e-02	2.66e-01	2.26e-01	2.12e-01	5.09e-01	-3.96e-01	5.64e-02	-2.14e-01	-2.66e-01	-1.79e-02	2.07e-01	4.84e-01	-1.44e-01
$\boldsymbol{v_2}$:	2.60e-02	-1.22e-01	-7.65e-02	-3.09e-01	-7.53e-02	-6.58e-02	-4.21e-01	-5.37e-01	3.52e-02	5.37e-01	-8.90e-02	2.25e-01	2.45e-01
$\boldsymbol{v_3}$:	-1.41e-01	4.96e-02	4.99e-01	-2.71e-02	-3.08e-01	1.70e-01	1.49e-01	-1.54e-01	-1.43e-01	-2.14e-01	-6.26e-01	3.16e-01	2.05e-03
$\boldsymbol{v_4}$:	9.17e-02	5.57e-02	-4.79e-01	5.28e-02	2.00e-01	-4.28e-01	2.87e-01	1.01e-01	6.61e-02	6.09e-02	-6.12e-01	-1.06e-02	2.39e-01
$\boldsymbol{v_5}$:	5.68e-02	-6.22e-02	-7.13e-02	6.79e-02	2.71e-01	1.56e-01	-3.23e-01	-3.81e-02	7.27e-01	-3.52e-01	-1.31e-01	3.00e-01	-1.42e-01
$\boldsymbol{v_6}$:	-4.64e-01	3.04e-01	-3.04e-01	-3.20e-01	2.86e-01	4.06e-01	2.79e-02	8.41e-02	-1.49e-01	1.98e-01	-1.46e-01	6.50e-02	-3.95e-01
$\boldsymbol{v_7}$:	8.32e-01	4.29e-02	2.57e-02	-1.63e-01	4.96e-02	1.87e-01	6.07e-02	1.89e-02	-1.09e-01	1.52e-01	-1.51e-01	-3.36e-03	-4.23e-01
$\boldsymbol{v_8}$:	1.14e-01	-4.24e-02	-1.17e-01	2.16e-01	1.96e-01	2.33e-01	-5.95e-01	2.59e-01	-5.01e-01	-2.03e-01	-1.70e-01	2.88e-02	2.99e-01
$\boldsymbol{v_9}$:	-1.17e-01	9.56e-02	2.37e-01	1.34e-01	-2.09e-01	-3.68e-01	-3.72e-01	5.34e-01	1.37e-01	3.99e-01	-1.49e-01	3.93e-02	-3.13e-01
$\boldsymbol{v_{10}}$:	-1.20e-02	-6.04e-01	-3.18e-02	-2.91e-01	5.62e-02	3.38e-02	2.28e-01	4.19e-01	-7.64e-02	6.59e-02	1.37e-01	5.30e-01	8.86e-02
$\boldsymbol{v_{11}}$:	-8.99e-02	-2.59e-01	4.82e-02	-5.22e-01	8.58e-02	-4.37e-01	-2.32e-01	-1.06e-01	-1.74e-01	-4.28e-01	-8.52e-02	-2.79e-01	-2.97e-01
$\boldsymbol{v_{12}}$:	-1.57e-01	-6.01e-01	-4.64e-02	5.24e-01	1.37e-01	7.81e-02	4.48e-02	-2.66e-01	-1.01e-01	1.84e-01	-1.66e-01	-1.64e-01	-3.76e-01
$\boldsymbol{v_{13}}$:	1.44e-02	7.94e-02	-5.39e-01	1.62e-01	-5.76e-01	-1.20e-01	-7.68e-02	-1.20e-01	-1.58e-01	-2.32e-01	1.27e-01	3.65e-01	-2.87e-01

5.2 cmc

Covariance Matrix

Table 7: Covariance Matrix

1.00e+00	5.40e-01	-4.87e-02	-4.32e-02	-4.98e-02	-1.73e-02	1.80e-01	2.97e-02	8.55e-02	-2.31e-04	-8.09e-02	-4.42e-03
5.40e-01	1.00e+00	-1.97e-01	-1.53e-01	2.64e-02	4.22e-02	-5.90e-03	3.50e-02	1.45e-03	1.21e-02	-8.62e-03	-4.91e-03
-4.87e-02	-1.97e-01	1.03e+00	5.12e-01	-8.43e-02	-2.74e-02	3.58e-01	-8.93e-02	1.85e-01	-3.90e-02	-1.29e-01	-1.69e-02
-4.32e-02	-1.53e-01	5.12e-01	6.66e-01	-5.19e-02	3.89e-04	2.85e-01	-6.17e-02	1.38e-01	-4.73e-02	-8.25e-02	-8.56e-03
-4.98e-02	2.64e-02	-8.43e-02	-5.19e-02	1.27e-01	1.08e-02	-6.83e-02	5.62e-03	-4.67e-03	-1.53e-02	1.86e-02	1.38e-03
-1.73e-02	4.22e-02	-2.74e-02	3.89e-04	1.08e-02	1.88e-01	-3.24e-02	2.08e-04	-1.21e-03	-5.80e-03	9.88e-03	-2.88e-03
1.80e-01	-5.90e-03	3.58e-01	2.85e-01	-6.83e-02	-3.24e-02	9.53e-01	-6.43e-02	1.30e-01	-1.55e-02	-1.12e-01	-3.13e-03
2.97e-02	3.50e-02	-8.93e-02	-6.17e-02	5.62e-03	2.08e-04	-6.43e-02	6.86e-02	-1.31e-02	3.09e-03	7.28e-03	2.72e-03
8.55e-02	1.45e-03	1.85e-01	1.38e-01	-4.67e-03	-1.21e-03	1.30e-01	-1.31e-02	2.09e-01	-8.55e-02	-1.18e-01	-5.43e-03
-2.31e-04	1.21e-02	-3.90e-02	-4.73e-02	-1.53e-02	-5.80e-03	-1.55e-02	3.09e-03	-8.55e-02	2.05e-01	-1.15e-01	-5.29e-03
-8.09e-02	-8.62e-03	-1.29e-01	-8.25e-02	1.86e-02	9.88e-03	-1.12e-01	7.28e-03	-1.18e-01	-1.15e-01	2.40e-01	-7.28e-03
-4.42e-03	-4.91e-03	-1.69e-02	-8.56e-03	1.38e-03	-2.88e-03	-3.13e-03	2.72e-03	-5.43e-03	-5.29e-03	-7.28e-03	1.80e-02

Eigenvalues

Table 8: cmc Data Eigenvectors

[0.0, 0.0, 0.1, 0.1, 0.2, 0.2, 0.3, 0.3, 0.4, 0.6, 1.6, 1.8]

Eigenvectors

Table 9: cmc Data Eigenvectors

v_1:	-8.50e-16	8.44e-03	2.50e-02	-1.17e-01	9.73e-02	1.32e-01	4.64e-02	-1.46e-01	-6.58e-01	2.98e-02	-6.88e-01	-1.70e-01
v_2 :	1.85e-16	7.23e-03	4.47e-03	7.25e-02	-1.12e-01	-1.29e-02	-2.20e-02	8.83e-02	6.56e-01	2.93e-01	-5.78e-01	-3.51e-01
v_3:	2.03e-15	2.39e-02	-6.87e-02	-8.07e-02	7.52e-02	6.16e-02	4.99e-01	2.66e-01	1.20e-02	4.79e-01	-1.38e-01	6.41e-01
v_4:	-5.25e-15	3.38e-03	-3.29e-02	-3.47e-02	-9.56e-02	2.50e-01	-6.70e-01	-4.24e-01	7.88e-02	2.55e-01	-9.52e-02	4.65e-01
v_5:	-8.10e-16	1.52e-03	-5.22e-02	-9.64e-01	-2.49e-02	-2.18e-01	-2.39e-02	-5.51e-02	1.03e-01	1.37e-02	4.40e-02	-6.44e-02
v_6:	7.80e-16	2.11e-02	-1.89e-02	2.42e-02	9.73e-01	-1.08e-01	-1.15e-01	-7.40e-02	1.32e-01	5.59e-02	3.52e-03	-2.69e-02
v_7:	1.91e-15	-5.47e-03	-5.10e-02	-3.10e-02	3.34e-02	1.03e-02	7.93e-02	3.96e-03	2.59e-01	-7.80e-01	-3.71e-01	4.21e-01
v_8:	-1.10e-15	-4.50e-02	-9.92e-01	6.33e-02	-2.25e-02	-3.49e-02	-1.97e-02	-2.25e-03	-4.39e-02	2.91e-03	2.63e-03	-7.52e-02
v_9:	5.00e-01	-3.08e-01	4.69e-02	1.50e-01	-8.29e-02	-7.48e-01	-1.01e-01	-6.85e-02	-9.44e-02	7.25e-02	-1.18e-01	1.46e-01
v_{10} :	5.00e-01	-2.84e-01	-1.20e-03	-1.23e-01	7.34e-02	3.96e-01	-3.05e-01	6.27e-01	-5.01e-02	-5.00e-02	4.33e-03	-3.50e-02
v_{11} :	5.00e-01	-2.72e-01	-9.38e-03	-3.56e-02	3.14e-02	3.73e-01	4.19e-01	-5.60e-01	1.51e-01	-6.65e-03	1.07e-01	-1.04e-01
v_{12} :	5.00e-01	8.64e-01	-3.63e-02	8.84e-03	-2.19e-02	-2.14e-02	-1.33e-02	1.91e-03	-6.33e-03	-1.59e-02	6.51e-03	-7.51e-03

5.3 Hypothyroid

Covariance matrix

Table 10: Covariance matrix

1.00e+00	-5.12e-02	-2.14e-01	-3.51e-02	-1.55e-01	5.29e-02	4.91e-03	4.81e-03	-2.04e-03	-6.78e-03	1.54e-02	-1.34e-02	-3.47e-03	6.57e-03	9.54e-03	-9.24e-03	-2.08e-03	-4.90e-03	-3.94e-03	-4.06e-04	-2.16e-02	-3.15e-02	-3.76e-02	3.79e-03	1.28e-01	-6.32e-02
-5.12e-02	1.00e+00	-1.49e-01	-2.60e-01	7.20e-02	-2.92e-01	-1.76e-02	6.27e-03	-1.71e-03	-1.12e-03	-4.13e-03	-2.31e-03	3.30e-03	-2.69e-04	6.89e-03	-2.32e-03	-3.91e-04	-1.35e-03	-2.55e-03	-5.09e-05	-3.82e-03	2.78e-03	-1.27e-02	-1.18e-03	-6.01e-03	1.71e-02
-2.14e-01	-1.49e-01	1.00e+00	5.08e-01	4.07e-01	3.09e-01	-3.22e-02	1.79e-03	-7.68e-04	8.42e-03	-1.48e-02	2.13e-02	-2.86e-03	1.50e-03	-1.16e-02	4.04e-02	5.41e-04	1.09e-03	1.53e-02	-2.55e-04	6.04e-03	3.81e-02	1.66e-02	1.36e-03	-1.30e-01	7.38e-02
-3.51e-02	-2.60e-01	5.08e-01	1.00e+00	4.28e-01	7.81e-01	-7.85e-02	7.02e-02	-4.12e-04	2.43e-03	-7.20e-03	2.04e-02	-2.43e-03	-9.50e-04	-1.96e-03	3.06e-02	-9.13e-04	-1.83e-03	8.94e-03	-4.15e-04	3.78e-04	3.24e-02	-7.38e-03	3.79e-03	-5.99e-02	3.11e-02
-1.55e-01	7.20e-02	4.07e-01	4.28e-01	1.00e+00	-1.73e-01	-9.89e-02	1.55e-02	1.39e-04	6.33e-03	-7.52e-03	3.95e-02	3.32e-03	1.25e-03	3.42e-03	1.79e-02	1.09e-03	3.33e-03	1.41e-02	1.10e-04	-2.89e-03	5.32e-02	-3.39e-03	1.98e-03	-8.28e-02	3.10e-02
5.29e-02	-2.92e-01	3.09e-01	7.81e-01	-1.73e-01	1.00e+00	-2.37e-02	6.14e-02	-2.86e-04	-1.87e-03	-4.04e-03	-1.84e-03	-3.70e-03	-1.97e-03	-4.64e-03	2.46e-02	-1.73e-03	-3.75e-03	1.78e-03	-5.00e-04	2.55e-03	2.21e-03	-7.10e-03	1.74e-03	-1.27e-02	1.59e-02
4.91e-03	-1.76e-02	-3.22e-02	-7.85e-02	-9.89e-02	-2.37e-02	2.73e-01	-1.35e-02	2.61e-03	-8.77e-04	1.15e-03	-4.23e-03	-1.57e-03	-1.73e-03	-3.59e-03	-6.72e-03	-1.95e-04	8.11e-04	-4.84e-03	1.95e-04	1.05e-02	-1.21e-02	2.16e-02	-3.98e-04	2.78e-02	-3.68e-02
4.81e-03	6.27e-03	1.78e-03	7.02e-02	1.55e-02	6.14e-02	-1.35e-02	1.08e-01	2.25e-04	-7.68e-05	-2.67e-03	3.93e-04	1.45e-03	2.58e-03	7.48e-03	-1.90e-03	-5.68e-05	-3.14e-04	-1.54e-03	-3.26e-05	-5.21e-03	-1.00e-03	-9.67e-03	5.84e-04	-2.63e-02	3.64e-02
-2.04e-03	-1.71e-03	-7.68e-04	-4.12e-04	1.39e-04	-2.86e-04	2.61e-03	2.25e-04	1.31e-02	-1.51e-04	2.79e-04	6.09e-04	7.89e-05	-2.07e-04	-8.23e-04	-3.03e-04	-6.33e-05	4.11e-04	-7.23e-05	2.62e-04	-6.47e-04	-3.94e-04	-1.36e-03	1.28e-04	-1.87e-04	1.81e-03
-6.78e-03	-1.12e-03	8.42e-03	2.43e-03	6.33e-03	-1.87e-03	-8.77e-04	-7.68e-05	-1.51e-04	1.13e-02	-4.44e-04	9.01e-04	-1.60e-04	8.68e-05	-4.42e-04	3.26e-03	-5.44e-05	-1.03e-04	-2.90e-04	-3.02e-06	-5.56e-04	9.87e-04	-1.17e-03	-1.18e-04	-3.13e-03	3.42e-03
1.54e-02	-4.13e-03	-1.48e-02	-7.20e-03	-7.52e-03	-4.04e-03	1.15e-03	-2.67e-03	2.79e-04	-4.44e-04	3.75e-02	-5.48e-04	-1.74e-05	-6.10e-04	1.29e-03	-1.65e-03	-1.86e-04	-3.51e-04	3.34e-04	-1.03e-05	-1.37e-03	-1.16e-03	-3.72e-03	5.43e-03	6.02e-03	-6.57e-03
-1.34e-02	-2.31e-03	2.13e-02	2.04e-02	3.95e-02	-1.84e-03	-4.23e-03	3.93e-04	6.09e-04	9.01e-04	-5.48e-04	1.39e-02	-1.97e-04	-2.20e-04	-6.07e-04	3.36e-03	-6.71e-05	1.38e-04	2.29e-03	-3.73e-06	-4.30e-04	6.21e-03	-1.17e-03	-1.45e-04	-3.85e-03	-1.04e-03
-3.47e-03	3.30e-03	-2.86e-03	-2.43e-03	3.32e-03	-3.70e-03	-1.57e-03	1.45e-03	7.89e-05	-1.60e-04	-1.74e-05	-1.97e-04	1.39e-02	4.53e-05	-3.42e-04	4.43e-04	-6.71e-05	-1.27e-04	-9.25e-05	-3.73e-06	-6.86e-04	1.13e-04	-1.44e-03	1.20e-04	-1.47e-03	2.67e-03
6.57e-03	-2.69e-04	1.50e-03	-9.50e-04	1.25e-03	-1.97e-03	-1.73e-03	2.58e-03	-2.07e-04	8.68e-05	-6.10e-04	-2.20e-04	4.53e-05	1.54e-02	1.42e-03	1.93e-03	-7.47e-05	-1.41e-04	-3.98e-04	-4.15e-06	-7.63e-04	-4.65e-04	-1.60e-03	-1.62e-04	-4.29e-03	6.52e-03
9.54e-03	6.89e-03	-1.16e-02	-1.96e-03	3.42e-03	-4.64e-03	-3.59e-03	7.48e-03	-8.23e-04	-4.42e-04	1.29e-03	-6.07e-04	-3.42e-04	1.42e-03	5.82e-02	1.14e-03	-3.09e-05	-5.59e-04	-1.31e-03	-1.65e-05	-6.40e-04	2.79e-04	-3.70e-03	1.54e-04	-1.63e-03	1.89e-03
-9.24e-03	-2.32e-03	4.04e-02	3.06e-02	1.79e-02	2.46e-02	-6.72e-03	-1.90e-03	-3.03e-04	3.26e-03	-1.65e-03	3.36e-03	4.43e-04	1.93e-03	1.14e-03	5.80e-02	2.30e-04	-5.66e-04	2.11e-03	-1.67e-05	-3.07e-03	2.38e-03	-5.11e-03	-1.19e-04	-9.54e-03	1.24e-02
-2.08e-03	-3.91e-04	5.41e-04	-9.13e-04	1.09e-03	-1.73e-03	-1.95e-04	-5.68e-05	-6.33e-05	-5.44e-05	-1.86e-04	-6.71e-05	-6.71e-05	-7.47e-05	-3.09e-05	2.30e-04	4.75e-03	-4.30e-05	-1.21e-04	-1.27e-06	5.63e-04	-1.42e-04	3.22e-03	-4.94e-05	-7.78e-04	-2.25e-03
-4.90e-03	-1.35e-03	1.09e-03	-1.83e-03	3.33e-03	-3.75e-03	8.11e-04	-3.14e-04	4.11e-04	-1.03e-04	-3.51e-04	1.38e-04	-1.27e-04	-1.41e-04	-5.59e-04	-5.66e-04	-4.30e-05	8.93e-04	3.57e-05	-2.39e-06	-1.75e-04	-2.68e-04	-6.57e-04	-9.32e-05	1.80e-04	8.38e-04
-3.94e-03	-2.55e-03	1.53e-02	8.94e-03	1.41e-02	1.78e-03	-4.84e-03	-1.54e-03	-7.23e-05	-2.90e-04	3.34e-04	2.29e-03	-9.25e-05	-3.98e-04	-1.31e-03	2.11e-03	-1.21e-04	3.57e-05	2.48e-02	-6.75e-06	-7.11e-04	1.90e-03	-2.07e-03	1.97e-06	-1.94e-03	2.12e-03
-4.06e-04	-5.09e-05	-2.55e-04	-4.15e-04	1.10e-04	-5.00e-04	1.95e-04	-3.26e-05	2.62e-04	-3.02e-06	-1.03e-05	-3.73e-06	-3.73e-06	-4.15e-06	-1.65e-05	-1.67e-05	-1.27e-06	-2.39e-06	-6.75e-06	2.65e-04	-1.29e-05	-7.87e-06	-2.71e-05	-2.74e-06	1.92e-04	-1.55e-04
-2.16e-02	-5.82e-03	6.04e-03	3.78e-04	-2.89e-03	2.55e-03	1.05e-02	-5.21e-03	-6.47e-04	-5.56e-04	-1.37e-03	-4.20e-04	-6.86e-04	-7.63e-04	-6.40e-04	-3.07e-03	5.63e-04	-1.75e-04	-7.11e-04	-1.29e-05	4.64e-02	-1.45e-03	3.72e-02	-5.04e-04	-1.07e-02	-2.45e-02
-3.15e-02	2.78e-03	3.81e-02	3.24e-02	5.32e-02	2.21e-03	-1.21e-02	-1.00e-03	-3.94e-04	9.87e-04	-1.16e-03	6.21e-03	1.13e-04	-4.65e-04	2.79e-04	2.38e-03	-1.42e-04	-2.68e-04	1.90e-03	-7.87e-06	-1.45e-03	2.88e-02	-3.04e-03	-3.07e-04	-8.14e-03	-1.73e-02
-3.76e-02	-1.27e-02	1.66e-02	-7.38e-03	-3.39e-03	-7.10e-03	2.16e-02	-9.67e-03	-1.36e-03	-1.17e-03	-3.72e-03	-1.17e-03	-1.44e-03	-1.60e-03	-3.70e-03	-5.11e-03	3.22e-03	-6.57e-04	-2.07e-03	-2.71e-05	3.72e-02	-3.04e-03	9.19e-02	-1.06e-03	-2.81e-02	-5.97e-02
3.79e-03	-1.18e-03	1.36e-03	3.79e-03	1.98e-03	1.74e-03	-5.98e-04	5.84e-04	1.28e-04	-1.18e-04	5.43e-03	-1.45e-04	1.20e-04	-1.62e-04	1.54e-04	-1.19e-04	-4.94e-05	-9.32e-05	1.97e-06	-2.74e-06	-5.04e-04	-3.07e-04	-1.06e-03	1.02e-02	-2.84e-03	-6.03e-03
1.28e-01	-6.01e-03	-1.30e-01	-5.99e-02	-8.28e-02	-1.27e-02	2.79e-02	-2.63e-02	-1.87e-04	-3.13e-03	6.02e-03	-3.85e-03	-1.47e-03	-4.29e-03	-1.63e-03	-9.54e-03	-7.78e-04	1.80e-04	-1.94e-03	1.92e-04	-1.07e-02	-8.14e-03	-2.81e-02	-2.84e-03	1.99e-01	-1.60e-01
-6.32e-02	1.71e-02	7.38e-02	3.11e-02	3.10e-02	1.59e-02	-3.68e-02	3.64e-02	1.81e-03	3.42e-03	-6.57e-03	-1.04e-03	2.67e-03	6.52e-03	4.89e-03	1.24e-02	-2.25e-03	8.38e-04	2.12e-03	-1.55e-04	-2.45e-02	-1.73e-02	-5.97e-02	-6.03e-03	-1.60e-01	2.43e-01

Eigen Values

Table 11: Hypothyroid Data Eigen Vectors

[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.1, 0.1, 0.1, 0.2, 0.4, 0.5, 0.8, 0.9, 1.5, 2.4]

Eigen Vectors

Table 12: Hypothyroid Data Eigenvectors

v_1:	0.00e+00	5.37e-04	1.28e-03	-5.63e-03	-2.99e-03	-4.10e-04	-5.70e-03	-2.57e-03	-1.89e-03	1.03e-02	7.18e-03	1.25e-02	5.72e-03	-2.76e-02	-4.58e-03	8.76e-03	3.00e-03	-1.36e-02	8.56e-02	-4.71e-02	1.00e-01	-2.84e-01	1.45e-01	-8.13e-01	4.53e-01	-1.06e-01
v_2:	-1.55e-15	2.41e-04	7.67e-04	-3.30e-03	2.54e-03	7.54e-04	-5.40e-03	-4.50e-03	1.45e-03	-1.43e-03	-4.91e-03	7.52e-03	-2.80e-02	6.64e-03	-4.29e-03	-6.61e-03	7.28e-03	-2.07e-02	1.40e-02	-1.17e-02	-8.00e-02	-1.08e-01	-8.24e-01	-2.98e-01	-3.85e-01	-2.52e-01
v_3:	-3.30e-15	1.42e-04	-1.00e-04	1.20e-03	3.10e-03	-4.64e-03	-1.59e-03	-3.38e-03	-5.04e-03	2.78e-03	5.57e-03	-1.25e-04	-1.71e-02	2.65e-02	4.74e-03	-2.93e-02	1.51e-02	8.11e-02	-5.58e-02	1.30e-01	-1.47e-01	-7.99e-01	7.64e-02	4.63e-02	-2.68e-01	4.81e-01
v_4:	-1.28e-14	6.86e-04	1.99e-03	-1.18e-02	1.10e-02	-2.17e-02	-3.12e-02	-2.26e-02	-2.42e-02	4.11e-04	-3.68e-02	2.02e-01	-6.34e-01	6.74e-02	-1.07e-02	-2.38e-02	2.51e-02	-2.54e-02	1.17e-02	-9.02e-02	-1.85e-02	3.17e-01	-1.70e-01	-1.78e-01	7.57e-02	6.09e-01
v_5:	6.71e-15	-6.25e-04	-2.29e-03	1.02e-02	-2.45e-02	1.45e-02	3.70e-02	2.25e-02	1.25e-02	-3.70e-03	1.83e-02	-1.49e-01	3.86e-01	2.19e-04	3.35e-02	4.55e-03	-2.98e-03	-4.25e-02	-1.38e-02	-1.12e-01	-2.39e-02	3.45e-01	2.85e-01	-4.30e-01	-5.85e-01	2.86e-01
v_6:	8.59e-15	-1.35e-04	1.12e-04	6.08e-03	-1.21e-02	2.34e-02	2.38e-02	1.93e-02	1.99e-02	-6.92e-03	2.76e-02	-1.93e-01	5.72e-01	-6.25e-02	1.18e-02	-5.74e-03	-1.50e-02	-7.67e-02	7.02e-03	-1.19e-04	-1.71e-02	8.22e-02	-4.18e-01	8.81e-02	4.51e-01	4.79e-01
v_7:	-8.37e-16	-4.10e-04	3.82e-03	2.14e-03	-2.58e-03	-1.33e-03	2.89e-03	1.29e-02	-5.30e-03	-4.58e-03	-1.50e-03	1.66e-02	1.02e-02	-3.94e-02	-2.33e-02	7.43e-03	-1.72e-02	-8.63e-03	-1.25e-01	-9.08e-01	-3.61e-01	-1.17e-01	2.22e-02	7.84e-02	5.42e-02	-4.97e-02
v_8:	8.14e-15	-3.38e-04	-4.47e-03	-2.56e-05	-1.21e-02	-7.14e-03	2.17e-04	8.65e-03	1.50e-02	1.88e-02	-9.90e-03	8.13e-03	5.52e-02	3.30e-02	-2.61e-02	1.83e-01	3.78e-02	9.57e-01	-2.08e-02	-8.25e-02	1.64e-01	5.87e-02	-5.88e-02	-2.01e-02	1.26e-02	3.53e-02
v_9:	1.09e-13	-2.04e-02	4.63e-05	7.45e-02	-2.19e-01	2.96e-02	1.34e-01	-9.52e-01	7.70e-02	1.15e-01	3.44e-03	-5.63e-03	1.88e-02	7.43e-03	-1.02e-02	5.44e-03	1.99e-02	-2.52e-03	-1.57e-02	-1.22e-02	2.17e-03	1.53e-03	1.55e-03	2.62e-03	-1.89e-04	-4.85e-05
v_{10}:	2.26e-14	-2.03e-07	6.24e-03	-5.99e-02	-3.58e-01	8.22e-01	-4.14e-01	5.54e-02	9.82e-02	2.73e-02	5.28e-02	1.41e-02	-2.37e-02	-1.44e-02	-4.47e-03	5.30e-02	-3.18e-02	-8.57e-03	-1.09e-02	-7.00e-04	7.20e-03	-5.05e-03	3.10e-03	3.63e-03	-6.85e-03	3.40e-03
v_{11}:	-4.23e-14	5.63e-04	-4.04e-04	-1.94e-02	-1.09e-01	-1.18e-01	-1.52e-01	1.20e-02	3.04e-02	-2.15e-02	1.86e-02	-3.75e-02	6.38e-02	4.95e-01	-8.28e-01	-9.88e-02	-1.74e-02	-2.40e-02	-2.49e-02	6.45e-03	-1.03e-02	8.03e-03	7.74e-03	-1.01e-02	1.11e-02	-7.39e-03
v_{12}:	-3.32e-14	1.95e-03	8.40e-03	2.06e-02	7.02e-01	-1.32e-02	-6.08e-01	-2.24e-01	1.74e-01	6.07e-02	-2.01e-02	1.26e-01	8.57e-02	-1.38e-01	-6.58e-02	5.85e-02	-3.12e-02	-5.25e-03	-8.39e-03	3.38e-03	-7.64e-03	1.42e-02	1.27e-02	-9.15e-03	-2.35e-02	1.53e-02
v_{13}:	-6.50e-16	6.44e-04	4.47e-03	-3.83e-02	2.79e-02	2.98e-02	-2.02e-01	-1.13e-01	-9.69e-01	8.52e-03	2.07e-02	-2.14e-02	4.26e-02	-1.16e-03	-5.81e-03	1.82e-02	3.73e-03	8.67e-03	-9.33e-03	-1.15e-04	9.97e-03	5.54e-03	-1.11e-03	5.09e-04	-4.32e-03	-1.62e-03
v_{14}:	1.94e-14	-4.40e-04	3.90e-03	-1.41e-02	1.62e-02	2.27e-02	-3.39e-02	-1.28e-01	1.63e-02	-9.88e-01	4.43e-02	-1.46e-02	-8.86e-03	1.12e-04	2.44e-02	1.94e-02	-4.21e-02	1.61e-02	-2.21e-03	-6.28e-03	2.46e-02	-7.83e-03	2.44e-03	-5.75e-03	6.65e-06	-1.30e-04
v_{15}:	5.65e-15	-2.03e-05	-1.57e-03	-9.14e-03	4.84e-03	1.20e-02	-1.14e-02	-2.12e-02	-8.39e-03	2.66e-02	-5.20e-02	-1.60e-02	-1.08e-02	-3.18e-02	6.46e-02	-6.20e-01	-7.63e-01	1.45e-01	-1.78e-02	-5.63e-03	3.07e-02	1.33e-02	-3.83e-03	-1.40e-02	-1.42e-05	-4.36e-03
v_{16}:	-1.19e-15	-2.73e-04	-9.06e-03	-8.24e-03	-1.28e-02	-5.36e-02	5.80e-02	5.76e-03	-3.78e-03	3.79e-02	2.63e-02	-1.54e-02	-3.53e-02	1.15e-01	-6.12e-03	7.44e-01	-6.33e-01	-1.25e-01	-5.73e-02	1.22e-02	2.79e-02	-2.86e-02	-1.04e-02	-3.85e-05	-9.68e-03	2.53e-02
v_{17}:	-5.37e-14	3.73e-04	9.98e-01	5.60e-03	-1.08e-02	-1.47e-02	-6.52e-03	1.84e-03	4.76e-03	3.68e-03	-1.93e-02	-4.35e-02	-1.09e-02	8.39e-05	3.28e-03	8.30e-03	-6.15e-03	5.55e-03	2.35e-02	2.27e-03	-5.31e-03	5.73e-04	1.77e-03	1.61e-03	-1.65e-03	-2.21e-04
v_{18}:	3.19e-14	1.90e-03	6.11e-03	-9.93e-01	4.47e-02	-2.38e-02	5.51e-02	-7.44e-02	3.80e-02	2.22e-02	-2.47e-03	-6.45e-03	1.63e-02	9.57e-03	1.80e-02	-3.28e-03	1.73e-02	-8.64e-04	-1.11e-02	-2.97e-03	-7.17e-04	1.46e-03	4.19e-03	3.38e-03	-3.94e-03	-2.63e-04
v_{19}:	7.44e-15	-1.50e-05	2.71e-03	-4.53e-03	-7.66e-02	4.59e-02	4.92e-02	1.52e-02	-3.21e-02	-5.25e-02	-8.89e-01	3.94e-01	1.68e-01	-5.05e-02	-6.45e-02	4.80e-02	-6.12e-05	-2.98e-02	-1.72e-02	1.66e-02	3.74e-03	-5.98e-03	5.51e-03	-3.32e-03	-8.63e-03	8.29e-03
v_{20}:	-4.90e-12	1.00e+00	-3.97e-04	3.43e-03	-5.83e-03	7.97e-04	4.10e-03	-1.88e-02	1.80e-03	1.75e-03	1.12e-04	-5.83e-04	9.04e-04	3.76e-04	5.52e-04	3.20e-04	2.40e-04	3.63e-04	-1.02e-03	-3.32e-04	-7.07e-04	3.93e-04	3.68e-04	3.29e-04	-2.54e-04	-2.37e-04
v_{21}:	-4.25e-14	5.76e-05	2.99e-02	5.04e-03	9.99e-03	2.24e-02	5.09e-02	1.49e-03	-1.44e-02	2.91e-03	3.08e-01	6.91e-01	2.48e-01	4.13e-01	2.21e-01	-3.47e-02	-2.10e-02	2.48e-03	3.65e-01	-2.60e-02	-7.79e-02	2.24e-03	3.22e-03	2.66e-02	-4.02e-03	2.72e-03
v_{22}:	4.47e-01	2.33e-04	1.41e-02	-2.90e-02	-2.67e-01	-1.89e-01	-6.28e-02	3.93e-02	-1.14e-03	-1.21e-02	2.39e-01	3.50e-01	5.82e-02	-6.29e-01	-3.10e-01	3.59e-02	-5.88e-02	1.08e-02	1.55e-02	5.28e-02	-4.17e-02	1.94e-02	6.86e-03	-3.72e-03	-3.74e-02	2.48e-02
v_{23}:	4.47e-01	3.05e-04	-4.26e-02	-1.29e-02	-8.22e-02	-1.08e-01	-1.92e-01	-2.24e-02	2.85e-02	1.88e-03	-2.08e-01	-3.49e-01	-8.11e-02	1.01e-01	1.02e-01	3.83e-02	-1.97e-02	3.86e-02	7.17e-01	-4.36e-02	-1.63e-01	-1.04e-02	1.86e-02	4.76e-02	-1.20e-02	1.91e-03
v_{24}:	4.47e-01	-2.76e-05	1.56e-02	3.32e-02	4.72e-01	4.85e-01	5.32e-01	-2.12e-02	-8.06e-02	-2.38e-03	1.80e-02	-1.49e-02	-2.80e-02	2.22e-02	-2.20e-01	-9.01e-03	-7.75e-03	9.84e-03	1.61e-02	3.74e-03	-6.29e-03	1.08e-03	1.31e-03	-4.72e-03	1.28e-03	1.79e-03
v_{25}:	4.47e-01	-9.07e-04	4.65e-03	5.55e-03	-6.20e-02	-8.78e-02	-1.40e-01	-2.18e-03	2.39e-02	-6.32e-03	-3.11e-02	-2.14e-03	2.43e-02	2.86e-01	2.46e-01	-1.21e-02	3.02e-02	8.47e-02	-5.14e-01	2.40e-01	-5.04e-01	1.04e-01	2.60e-02	-9.99e-02	1.15e-01	-6.99e-02
v_{26}:	4.47e-01	3.97e-04	8.23e-03	3.16e-03	-6.09e-02	-1.00e-01	-1.37e-01	6.53e-03	2.93e-02	1.89e-02	-1.70e-02	1.59e-02	2.66e-02	2.19e-01	1.82e-01	-5.32e-02	5.59e-02	-1.44e-01	-2.35e-01	-2.53e-01	7.16e-01	-1.14e-01	-5.28e-02	6.07e-02	-6.73e-02	4.14e-02

References

[1] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[2] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[3] Julio-Omar Palacio-Niño and Fernando Berzal. Evaluation metrics for unsupervised learning algorithms. *arXiv preprint arXiv:1905.05667*, 2019.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.