Exploratory Data Analysis - Written Exam

André Plancha
andre.plancha@hotmail.com
MALMÖ UNIVERTITY
Spring 2025

Computer Science: Applied Data Science

March 29, 2025

1. Due to large sizes of data sets in practice one need to do dimension reduction thereafter study the pattern by various clusterings techniques. Dimension reduction means lost of certain information whence less accuracy of clustering results in general. In this question, we consider the Iris data set as four dimensional data set with the ground truce label: setosa, versicolor, virginica and use the k-means for clustering.

a) First determine three clusters of Iris data set by k-means, and even compute the percentage of correct classified observations.

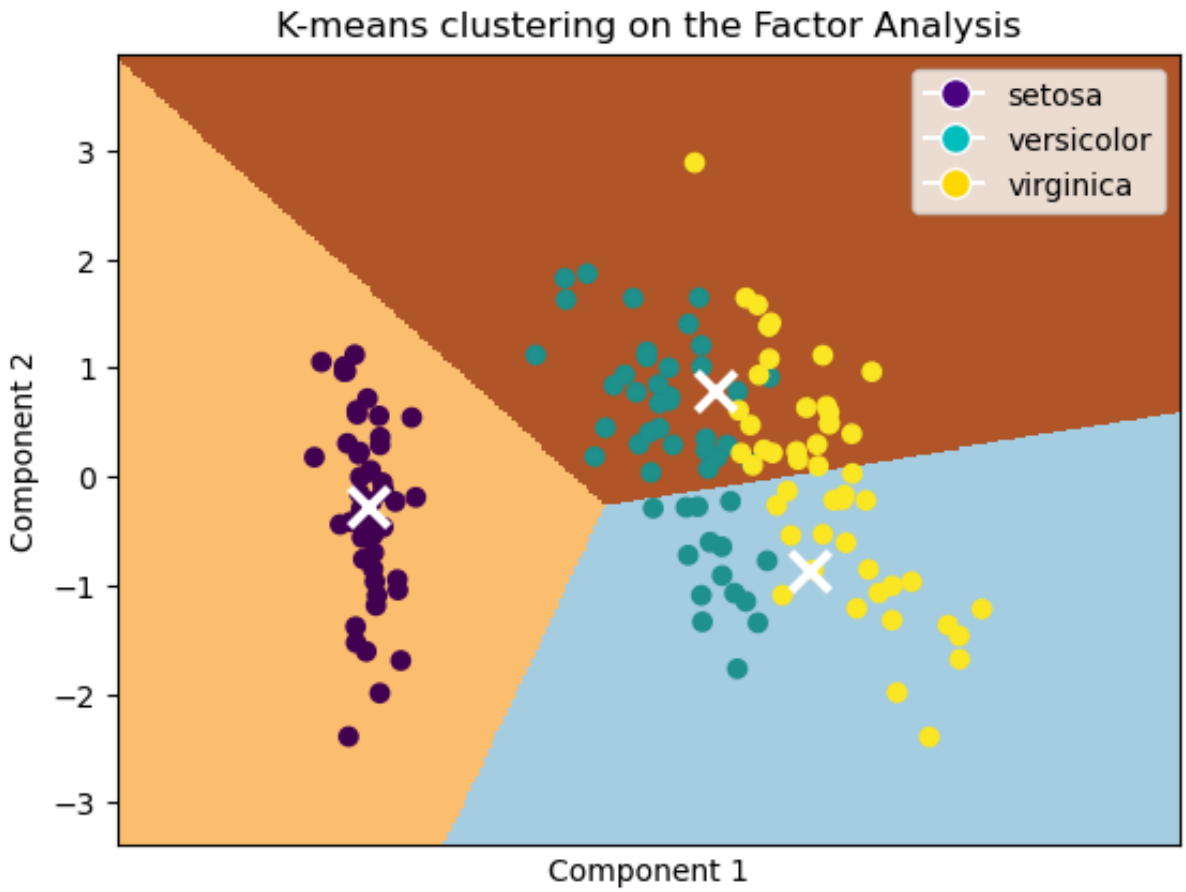
K-means clustering is an iterative algorithm that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. Using scikit-learn's `KMeans`, the following was obtained:

	Cluster 1	Cluster 2	Cluster 3
setosa	0 (0%)	50 (100%)	0 (0%)
versicolor	47 (77%)	0 (0%)	3 (8%)
virginica	14 (23%)	0 (0%)	36 (92%)
Total	61	50	39

From this, we can see that the clusters obtained are able to separate the classes quite well, with 100% of setosa on one cluster, 77% of versicolor on another, and 92% of virginica on the last one. This leads us to a total of 82% of correct classified observations.

b) Now apply first the factor analysis method to reduce the Iris data set to two dimensional, then determine three clusters of the reduce data set by k-means. What is then the percentage of correct classified observations? Even visualize the reduced data set with original label respectively by the clustering.

Factor analysis uses the correlation structure amongst observed variables to model a smaller number of unobserved, latent variables known as factors. Using scikit-learn's `FactorAnalysis` and scikit-learn's `KMeans`, the following was obtained (the color in the background represents the cluste division):



	Cluster 1	Cluster 2	Cluster 3
setosa	0 (0%)	50 (100%)	0 (0%)
versicolor	15 (38%)	0 (0%)	35 (57%)
virginica	24 (62%)	0 (0%)	26 (43%)
Total	39	50	61

From this, we can clearly see that the algorithm was able to separate setosas from the other classes, but it had a hard time separating versicolors from virginicas. This leads us to a total of $\approx 72.67\%$ of correct classified observations. This is a decrease in the accuracy of the classification, which is expected when reducing the dimensionality of the data.

2. Consider the minimum spanning trees method (MST) and lung cancer data set `LungA`.

a) First apply MST to find two clusters, then find the percentage of correct clustered genes using the original label of cancer cell: `Normal, Cancer (Small-cell lung carcinomas, Nonsmall cell lung carcinomas)` as ground truth.

As I interpreted it, this questions asks us to:

1. Transform `LungA` into a undirected weighted complete graph, where the weight of the edge between two nodes is the euclidean distance between them;

2. Transform that graph into its minimum spanning tree, using for example Kruskal's algorithm;
3. Remove the longest edge from the tree, thus dividing it into two clusters;
4. Compare the clusters obtained with the original labels.

`LungA` is a dataset that seems to have 203 observations and 3312 unnamed numerical features, and each obsevration has a label between `AD`, `SQ`, `COID`, `NL`, and `SCLC`. Grouping `AD`, `SQ`, `COID` and `SCLC` as "Cancer", using the described steps above and using SciPy's `minimum_spanning_tree` and `connected_components`, the following was obtained:

	Cluster 1	Cluster 2
Cancer	185 (92%)	1 (100%)
Normal	17 (8%)	0 (0%)
Total	202	1

From this, we can see that the algorithm was not able to separate the normal cells from the cancer cells, since the smaller cluster has only one Cancer observation. Because of the nature of this algorithm, we can conclude that this observation might be an outlier, showing the method's robustness in outlier detection. Assuming cluster 1 is the cancer cluster, the percentage of correct classified observations is 91.13%.

b) Apply MST once again to find three clusters, then compare the correct clustered genes using the original label: `Nonsmall cell lung carcinomas, Normal, Small-cell lung carcinomas` as ground truth.

This time, `AD`, `SQ` and `COID` were grouped as "Nonsmall cell lung carcinomas", `SLC` as "Small-cell lung carcinomas", and `NL` as "Normal". Using the same steps as before, the following was obtained:

	Cluster 1	Cluster 2	Cluster 3
Nonsmall cell lung carcinomas	178 (89%)	1 (100%)	1 (100%)
Normal	17 (8%)	0 (0%)	0 (0%)
Small-cell lung carcinomas	6 (3%)	0 (0%)	0 (0%)
Total	201	1	1

The same conclusion can be drawn from this result, where the algorithm was not able to separate the observations based on the labels given, reinforcing the idea that the algorithm is robust to outliers. Aditionally, it's important to note that, since the method is a divise clustering method, cluster 3 is a subset of the previous cluster 1, and increasing the number of clusters will result in this cluster 1 being divided into smaller clusters.

c) Finally, test to find five clusters comparing the correct clustered genes using the label: `AD, COID, SQ, NL, SCLC` as ground truth. Even comment on these clustering results.

This time, `AD`, `COID`, `SQ`, `NL` and `SCLC` were used as labels. Using the same steps as before, the following was obtained:

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
AD	138 (70%)	1 (100%)	0 (0%)	0 (0%)
COID	16 (8%)	0 (0%)	3 (100%)	1 (100%)
NL	17 (9%)	0 (0%)	0 (0%)	0 (0%)
SMCL	6 (3%)	0 (0%)	0 (0%)	0 (0%)
SQ	21 (11%)	0 (0%)	0 (0%)	0 (0%)
Total	198	1	3	1

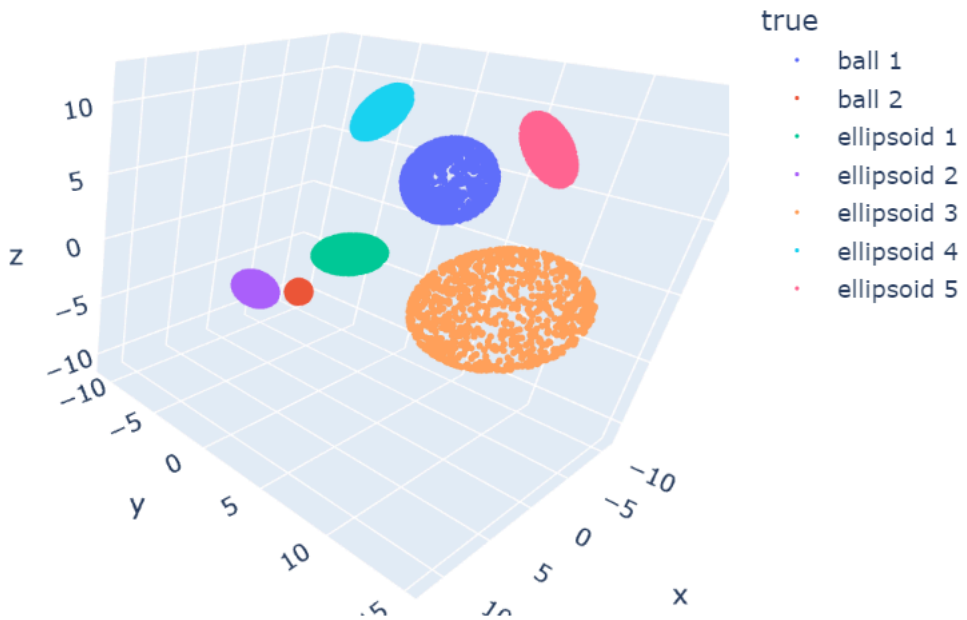
As in a) and b), the algorithm was not able to separate the observations based on the labels given. This might also be because there's a big imbalance between the number of `AD` observations, which make up a grand majority of the dataset, making the division difficult to obtain from the data.

3. Model based clustering in chapter 6 (section 6.5) is based on geometrical properties of clusters, e.g. balls and ellipsoids in 3-dim. In this question, you are asked to randomly generate data sets to check the capability of MBC: two balls of different sizes, two ellipsoids of different size with symmetry axis parallel with coordinates axis, three ellipsoids also different sizes with arbitrary symmetry axis.

a) Randomly generate those sub data sets so that they are disjoint (far away from each other), then apply model based clustering to fin clusters even compare the estimated clusters with original labels.

For this question, 1000 points of every shape were generated, with the following properties:

	x_c	y_c	z_c	x_r	y_r	z_r	α	β	γ
Ball 1	10	10	10	2	2	2	0°	0°	0°
Ball 2	−10	−10	−10	1	1	1	0°	0°	0°
Ellipsoid 1	0	0	0	3	2	1	0°	0°	0°
Ellipsoid 2	10	0	0	1	1.5	1	0°	0°	0°
Ellipsoid 3	0	10	0	4	5	1	30°	45°	30°
Ellipsoid 4	0	0	10	2	2	1	60°	0°	0°
Ellipsoid 5	0	10	10	3	1	1	60°	20°	45°



Model based clustering is a clustering method based on a finite mixutre probability model, using the EM algorithm

b) Randomly generate those sub data sets so that they are overlap partly, then apply model based clustering to find clusters even compare the associated clusters with original labels.

4. Find the datasets about inflation, unemployment in member countries of European union during the last ten years (2015 - 2024). Your dataset should contains at least 20 countries. Denote the inflation dataset by EUin and the unemployment dataset by EUun and EU = [EUin,EUun].

Hint You may check the database at the websites: https://european-union.europa.eu or https://ec.europa.eu/eurostat/data/database or

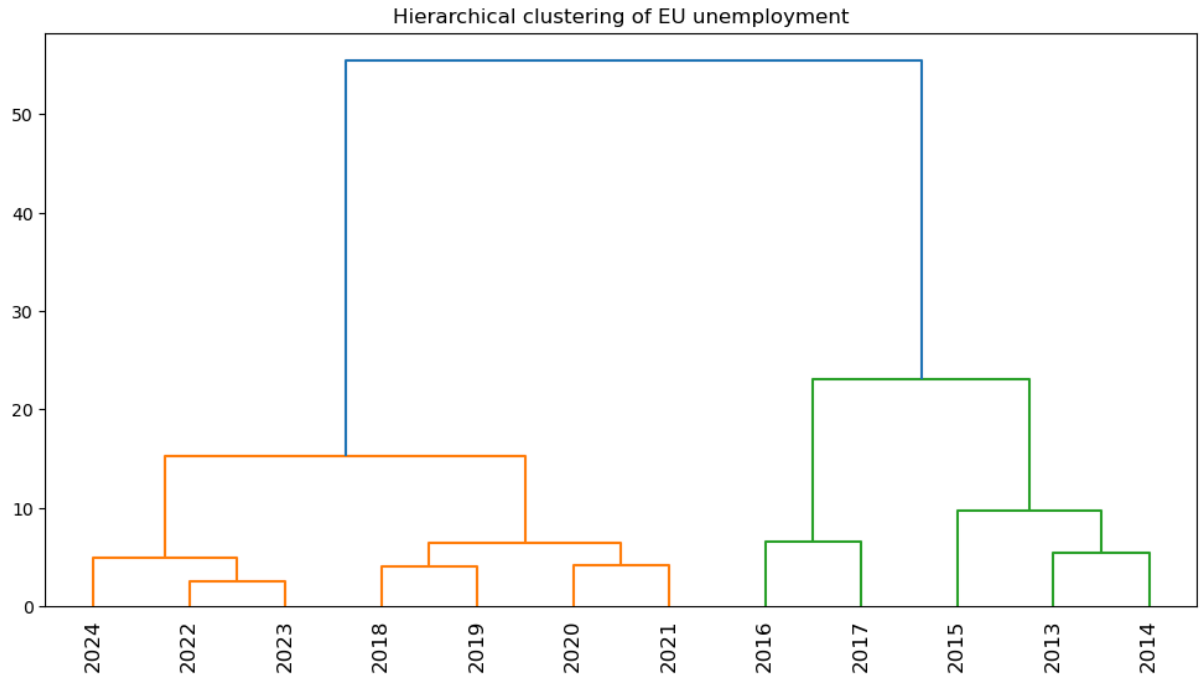
a) Study the unemployment pattern by looking at clusters of EUun using one proper hierarchical methods and evaluate the result by Silhouette plot.

EU_{un} collected is a 12x26 (12 years, 26 EU countries) table with the following structure:

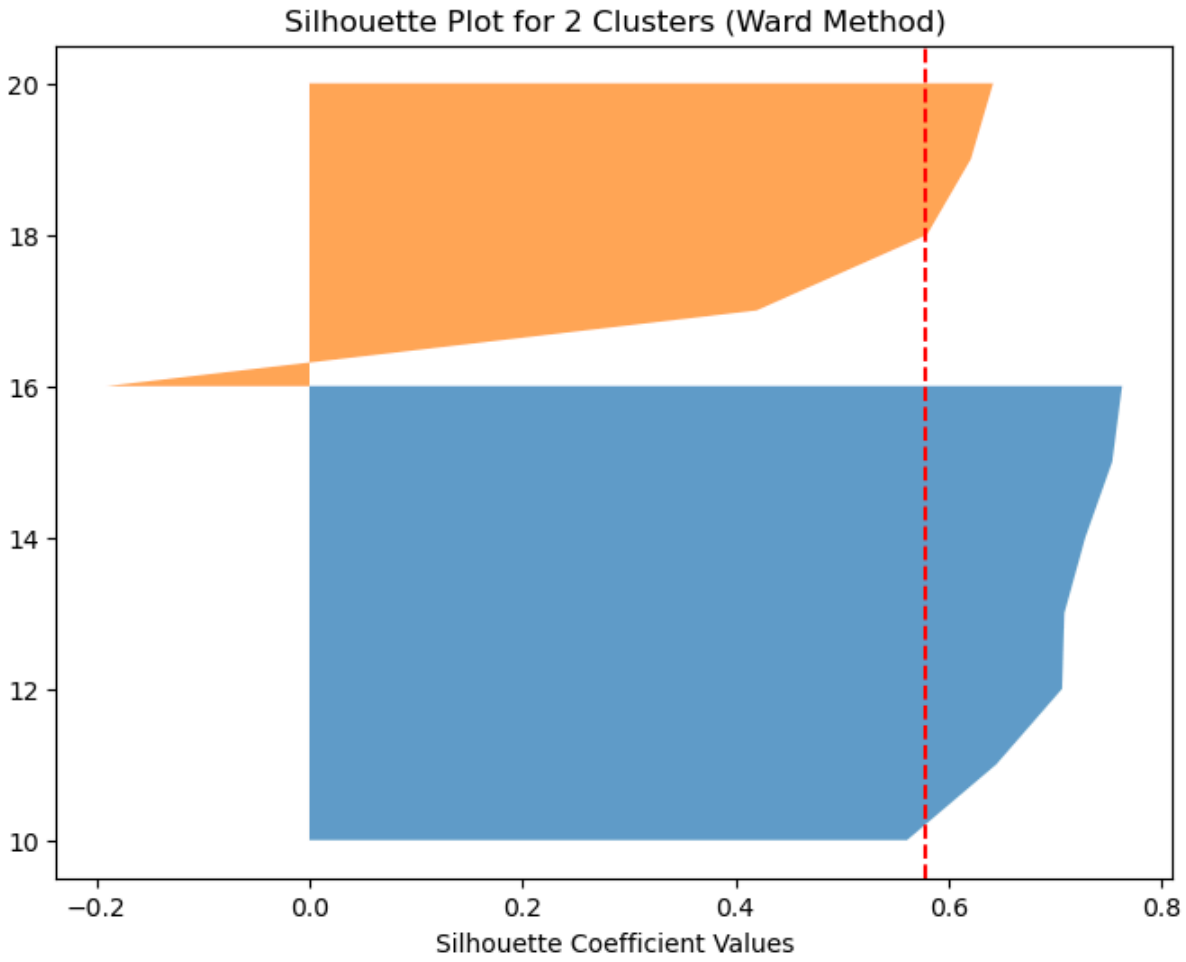
	Belgium	Bulgaria	...	Sweden
2013	8.6	13.9	...	8.1
2014	8.7	12.4	...	8.0
2015	8.7	10.1	...	7.5
...	...	...	...	...
2024	5.7	4.2	...	8.4

The full dataset is available on github.com/notPlancha/eda-exam/tree/main/data

Hierarchical clustering is a method of clustering analysis that builds a hierarchy of clusters, represented as a tree-like structure called a dendrogram. Using SciPy's `linkage` with Ward's method, the following dendrogram was obtained:



We can see that the method found a clear divide between years <= 2017 and > 2017 in the data, based on the distance between this divide. By defining these 2 as clusters, the following silhouette was produced:

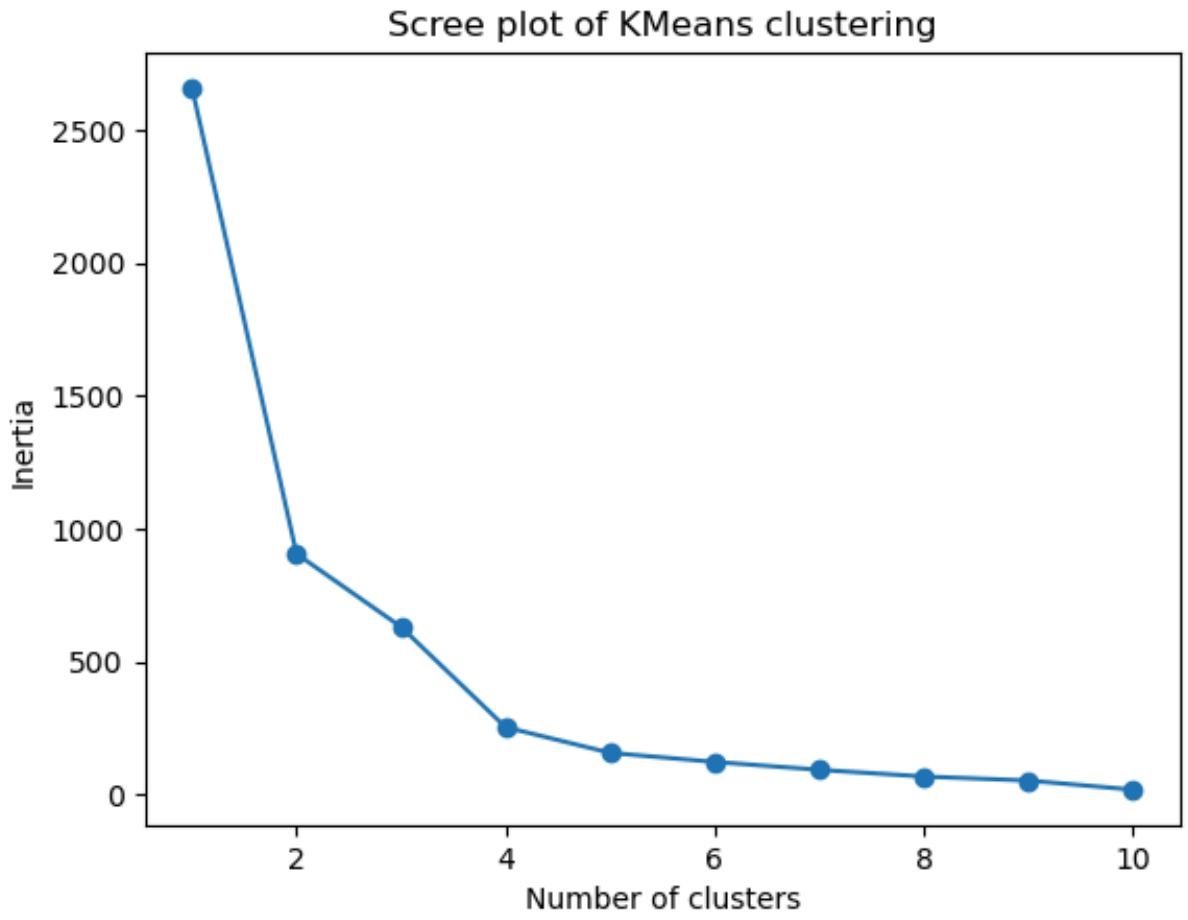


Looking at the silhouette plot, we can see that the clusters are well defined, with a silhouette score close to 0.6. There seems to be a single year that perhaps could be better in the other cluster, but overall the clusters are well defined.

b) Study the joint pattern of unemployment and inflation by looking at clusters of EU using k-means method and evaluate the result by using Dunn index.

EU collected is a 12x52 (12 years, 26*2 EU countries) table. EU_{in} is also available on the same link as before.

To determine the number of clusters that should be used with the k-means algorithm, a scree plot was produced:



By the plot and the elbow method, it seems that 4 clusters should be used. Using scikit-learn's `KMeans` with 4 clusters, the following Dunn index was obtained:

$$DI_m \approx 0.68$$

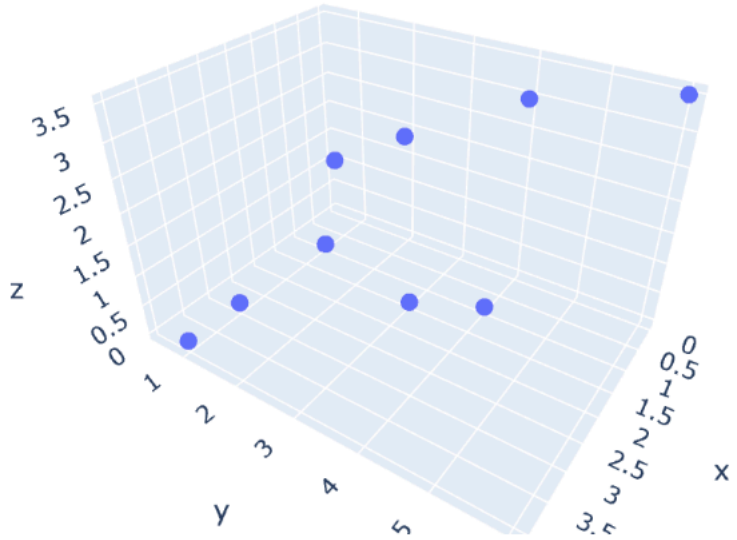
The Dunn index is a measure of cluster separation and cluster compactness representing the ratio between the minimum inter-cluster distance and maximum intra-cluster distance: the larger the index, the better the clustering. Altough the metric is better as a comparasion between cluster results, we can say that since the value is smaller than 1, the clustering is not optimal, but still good.

5. Consider the dataset

$$X = \begin{pmatrix} 3 & 1 & 0.1 \\ 1 & 4 & 0.02 \\ 1 & 1 & 0 \\ 4 & 1 & 0.1 \\ 1.5 & 3 & 0 \\ 0.12 & 2 & 1.9 \\ 0 & 6 & 3.5 \\ 0.03 & 0.5 & 1 \\ 0.1 & 4 & 3 \end{pmatrix}$$

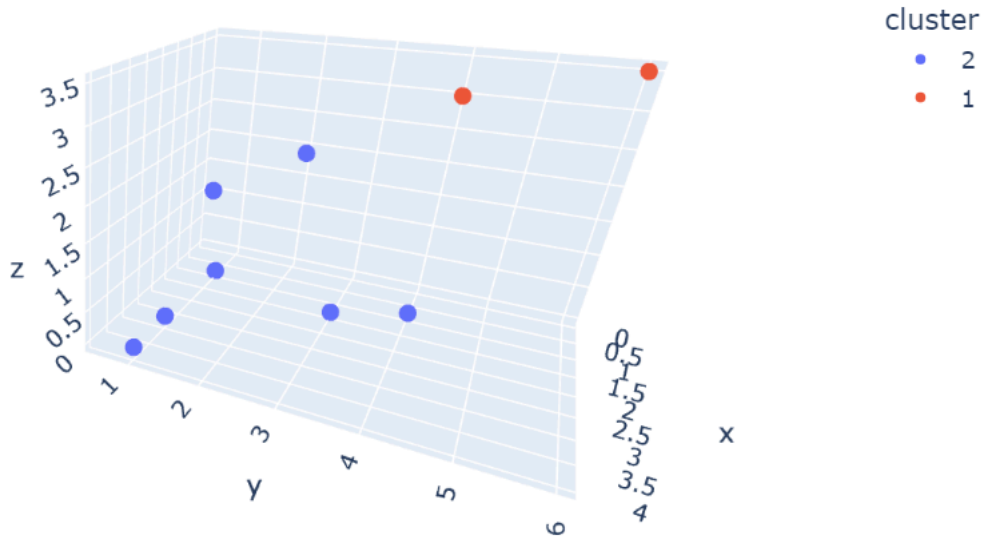
a) Visualize the dataset X via scatter3.

Using plotly's `scatter_3d`, the following was obtained:



b) It's obvious that X contains two clusters: one is on xy-plane and another one is on the yz-plane. Please check if hierarchical Ward's method, spectral method are able to recover these two clusters.

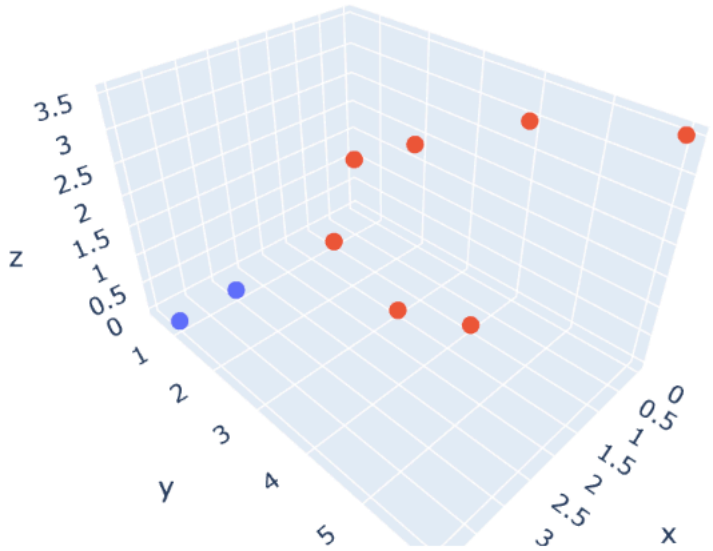
Ward's method is a method of hierarchical clustering that minimizes the within-cluster variance. Using SciPy's `linkage` with Ward's method, and reuqest 2 clusters, the following was obtained:



We can see from the plot that the method was not able to recover these 2 clusters, with 75% of correct classified observations (assuming the xy-plane is cluster 2). This is expected,

since, visually, the variance in the yz-plane seems to be higher than the variance in the xy-plane, and some points between the 2 are very close by.

Spectral clustering is a method that uses the eigenvalues of a similarity matrix to reduce the dimensionality of the data before clustering in a lower dimensional space. Using scikit-learn's `SpectralClustering` with 2 clusters, the following was obtained:



Unexpectedly, the spectral method was also not able to recover the clusters, with the same precision as Ward's method (assuming the yz-plane is cluster 0).

6. Consider again the dataset X given in the previous question.

a) Determine the non-negative factorization matrices W, H of X, i.e., $X = WH$, by Multiplicative update algorithm (MULT).

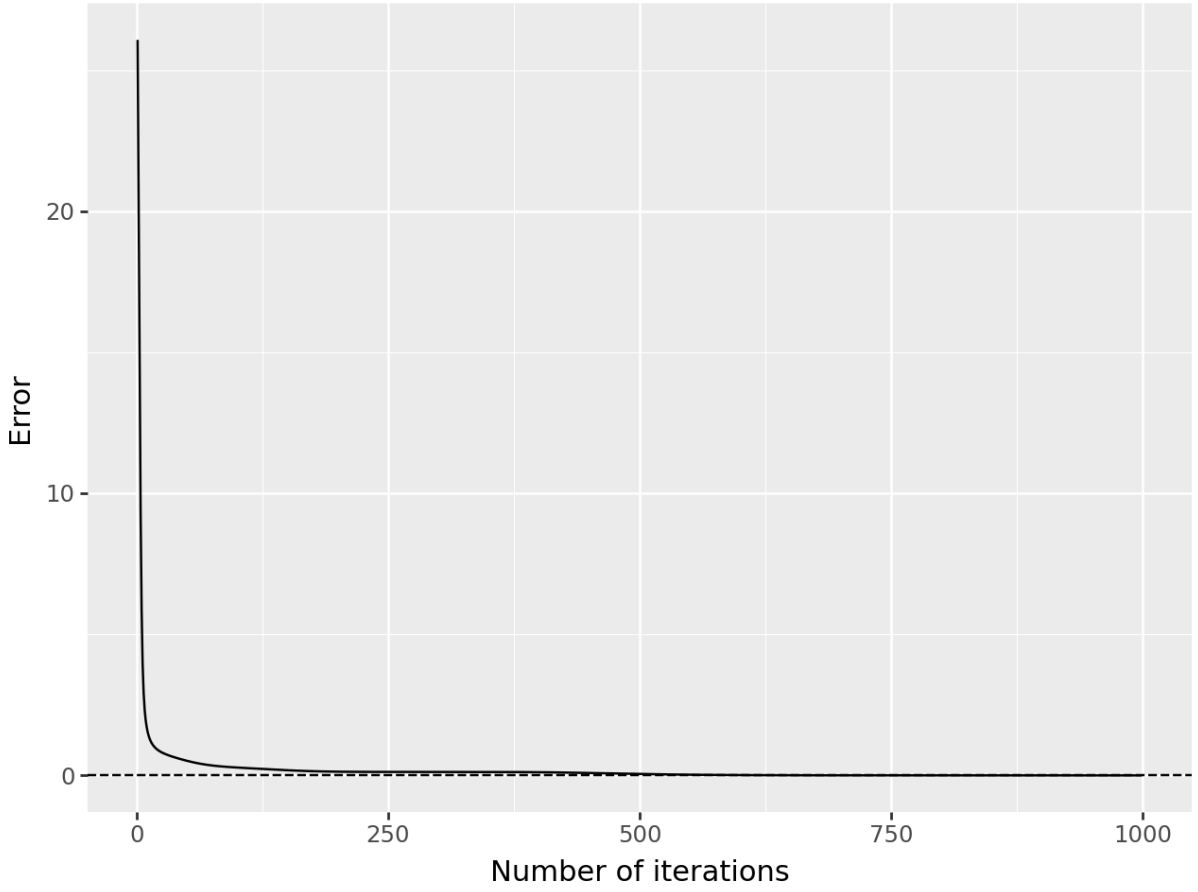
Non-negative matrix factorization (NMF) is a group of algorithms in multivariate analysis and linear algebra where a matrix X is factorized into two matrices W and H, with the property that all three matrices have no negative elements. The multiplicative update algorithm is a method to solve this factorization. Using scikit-learn's `non_negative_factorization` and the *mu* solver, the following matrices were obtained after 50 iterations:

$$W \approx \begin{pmatrix} 0.03 & 0.92 & 0.08 \\ 0.02 & 0.21 & 1.06 \\ 0 & 0.29 & 0.22 \\ 0.01 & 1.24 & 0.03 \\ 0.01 & 0.4 & 0.75 \\ 0.73 & 0.03 & 0 \\ 1.48 & 0 & 0.48 \\ 0.27 & 0 & 0 \\ 1.29 & 0.03 & 0.05 \end{pmatrix}$$

$$H \approx \begin{pmatrix} 0 & 3 & 2 \\ 3 & 1 & 0 \\ 0 & 4 & 0 \end{pmatrix}$$

b) In this question, we study the convergence of MULT. Let the maximum number of iterations be K and the error function f be $\|X - WH\|^2$, then f is a function of K. Modify the Matlab EDA toolbox routine `nnmf` or any your Python library such that MULT terminates whenever the number of iterations has reach the maximum number of iterations K.

i) Plot the error term f as an function of K.

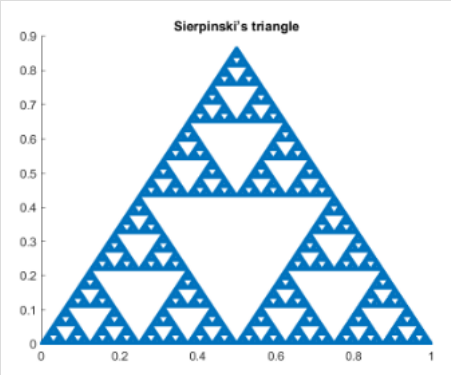


ii) For what integers K is the error f less than the tolerance $\varepsilon = 10^{-4}$?

In this case, $f(K)$ is smaller than 10^{-4} for all $K \geq 834$, altough the error seems to converge way before that. Regardless, $f(K)$ is always decreasing.

7. In the study of dimension reduction by Principle Component Analysis (PCA), one of PCA is based on covariance matrix, and another one is based on correlation matrix. The covariance matrix based PCA has a property: all the PC:scores are uncorrelated, i.e., the correlation coefficient between any pair of PC:scores is zero. Please give a short proof of this property.

8. Consider the Sierpinski triangle, which is a fractal set and is of Hausdorff dimension $\frac{\log_3}{\log_2}$, roughly 1.585.



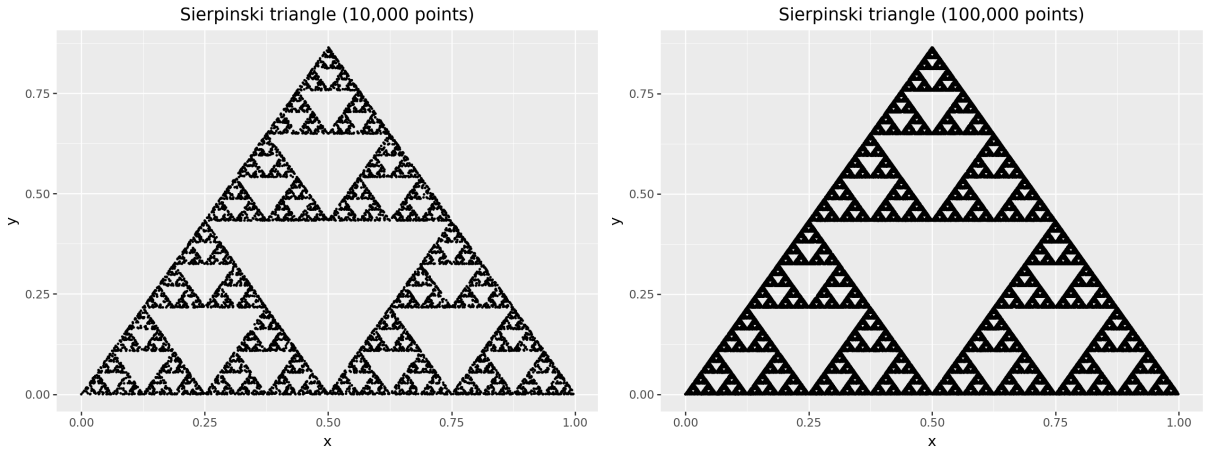
a) Please make samplings of points from the above Sierpinski triangle of sizes: a) 10000 points, b) 100000 points.

The Sierpinski triangle is a fractal set that can be generated by repeatedly removing the central triangle of an equilateral triangle. These points can be iteratively generated using the following rules, which are selected randomly with equal probability[1]:

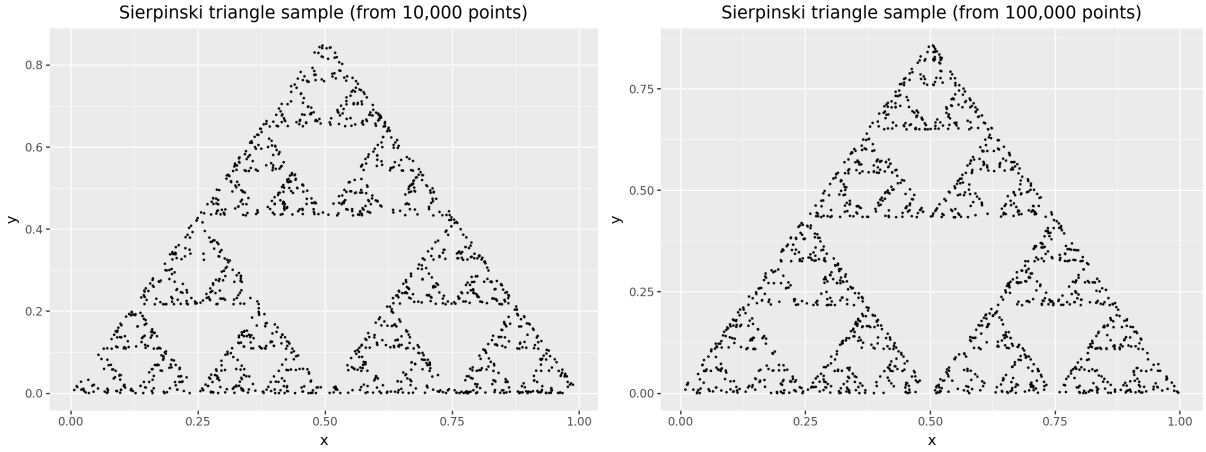
$$\begin{cases} x = 0.5x & \wedge\ y = 0.5y \\ x = 0.5x + 0.25 & \wedge\ y = 0.5y + \frac{\sqrt{3}}{4} \\ x = 0.5x + 0.5 & \wedge\ y = 0.5y \end{cases}$$

The 10,000 points and 10,000 generated look like this:

[1] obtained from https://www.mathworks.com/matlabcentral/answers/2180-plotting-sierpinski-s-triangle, which was the same source of Paulo Silva's code



From these, a sample of 2000 points was taken.

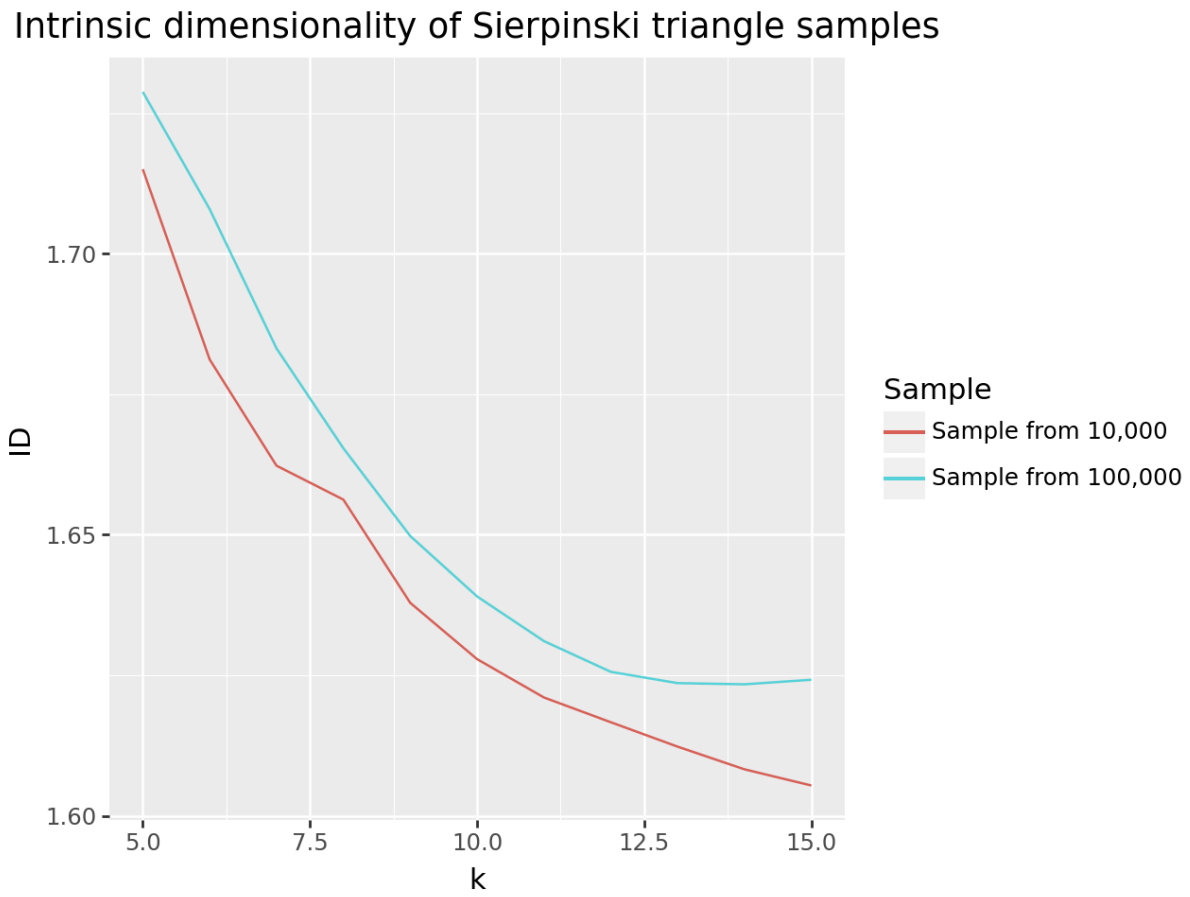


b) Estimate the intrinsic dimensionality of the above Sierpinski triangle by the nearest neighbor method `idpettis`.

The intrinsic dimensionality of a dataset a number that represents the estimate of independent components needed to represent the data. `idpettis` is a method that uses the Pettis, Bailey, Jain and Dubes algorithm[2] , based on the nearest neighbor distances. Using this method, the intrinsic dimensionality of the sampled points is ≈ 1.72 and 1.73, respectively. This means that for both samples, the intrinsic dimensionality is close to 2 dimensions, which is expected since the Sierpinski triangle is a 2D object.

c) The number of neighbors K in the above EDA toolbox `idpettis` is by default 5, study the sensitivity of the above estimations with respect to K. Plot these estimates as function of K, where $5 \leq K \leq 15$ and even comment on the behaviors of these curves.

[2]Pettis, K. W., T. A. Bailey, A. K. Jain, and R. C. Dubes. 1979. "An intrinsic dimensionality estimator from near-neighbor information," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1:25-37.



From the plot, we can see that the intrinsic dimension decreases as K increasing, arguably starting to plateau around $K = 12$. The intrinsic dimension of the sample from 100,000 points is always higher than the one from 10,000 points. Finally, the Intrinsic dimension of the samples only ranges between 1.6 and 1.8, suggesting that the algorithm, regardless of K, is able to estimate the intrinsic dimension of the Sierpinski triangle.