**ML II unsupervised learning, agents: project**

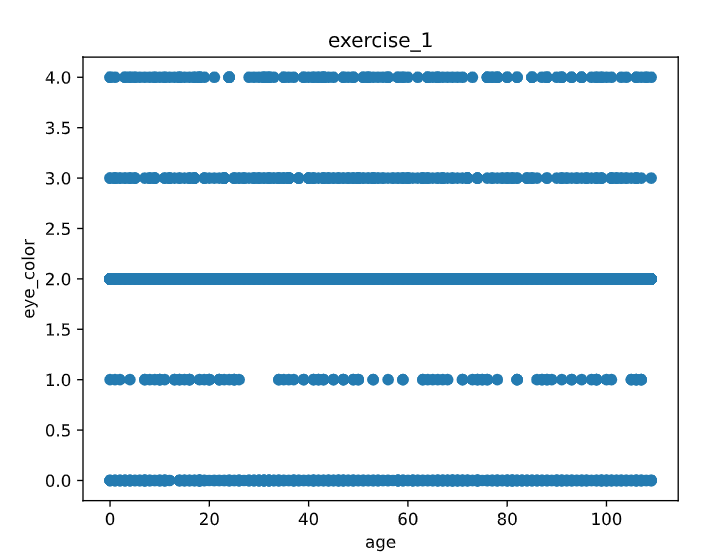
*Part 1: data distribution and the law of large numbers*

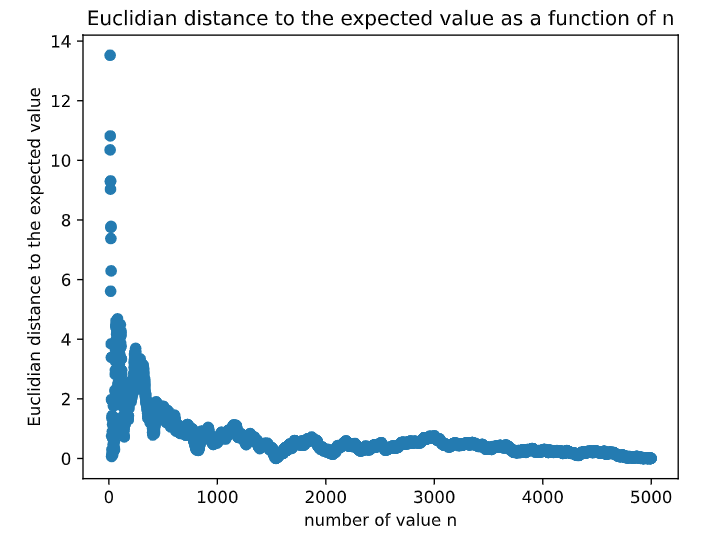
I spent a lot of time trying to figure out what was exactly the matter of the exercise. I wanted first to take as parameters the age of a population and their mass. But I quickly got lost in all random numbers, so I decided change mass parameter to eyes color following real percentage with far less different possible values.

It got better after this change, and I finally understood that goal was to demonstrate law of large numbers by checking difference between expected value and empirical mean and how it converges when size of data is large.

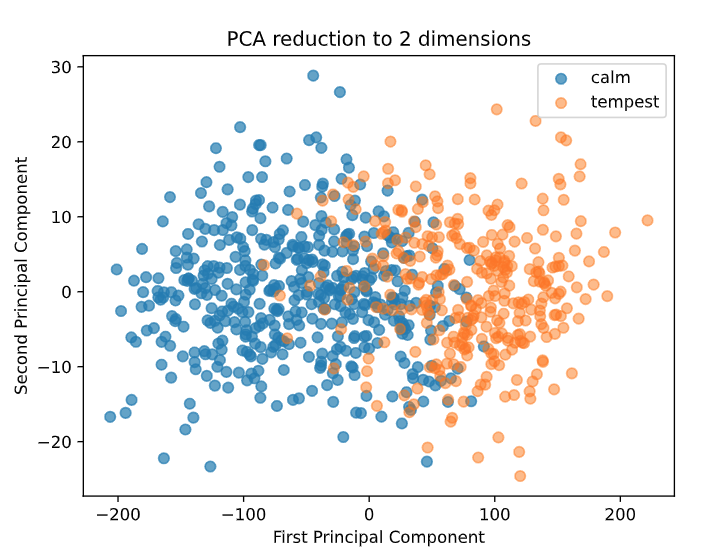
I just got issue on converging around 1 instead of 0 cause I created my array of eyes color with numpy.random.randint() instead of numpy.random.choice() so my custom probabilities for each value wasn’t take into account.

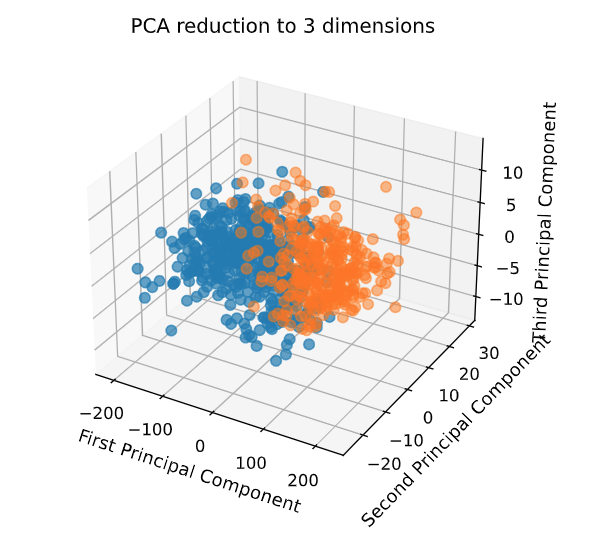
After this fix everything went well.





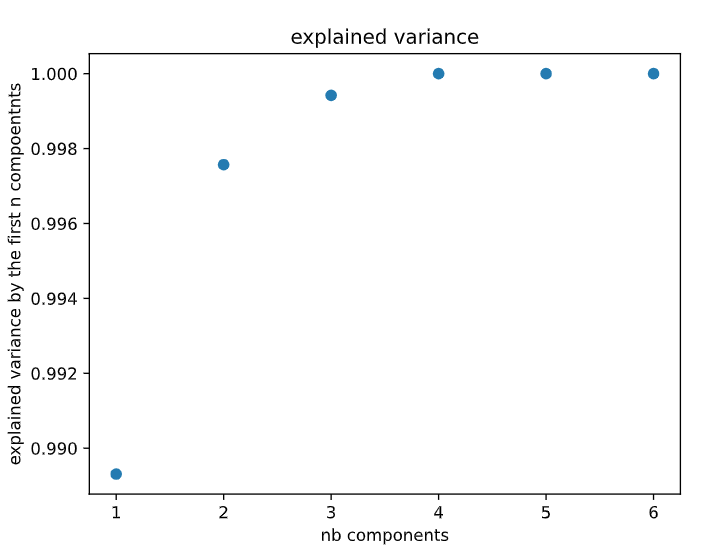
*Part 2: meteorological data: dimensionality reduction and visualization*





Which dimension, between 2 and 3, seems to allow to predict the label based on the projected components only?

A 2-dimensional reduction seems to allow to predict the label based on the projected components only. In, fact, we can figure out that this dataset is reductant.



*Part 3: Company clustering customers*

I started this project by choosing KMean with kneed metric to get a simple base. I wanted to then use Hierarchical clustering as second clustering method. I soon realized Kneed couldn’t be used for Hierarchical clustering, so I used silhouette score instead.

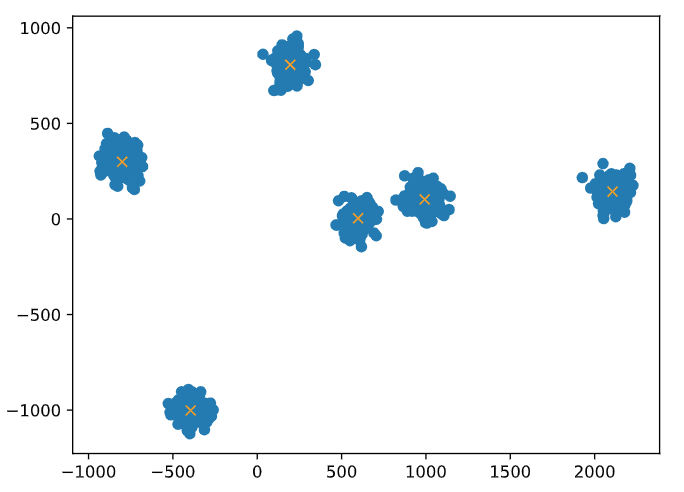
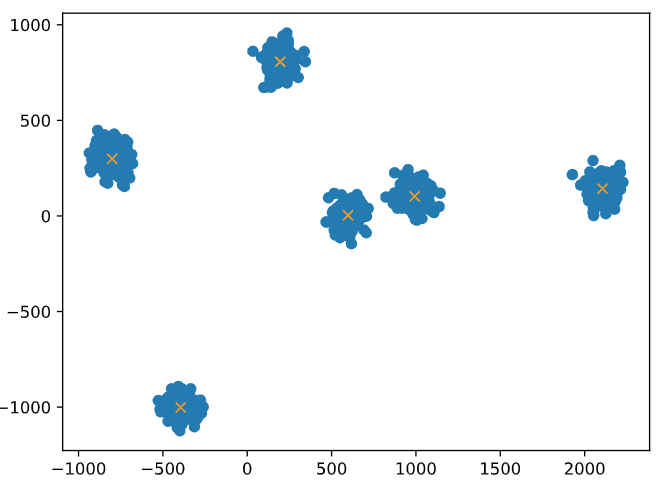
I also chosen Agglomerative Clustering as Hierarchical clustering because it was easy to implement thanks to scikit-learn. For metrics I decided to use Euclidian metric for KMean and try to use cosine metric for Agglomerative clustering. I wanted to try cosine as values was sometimes far from one to another and see if cosine metric could help minimize this distance and have a purpose doing it.

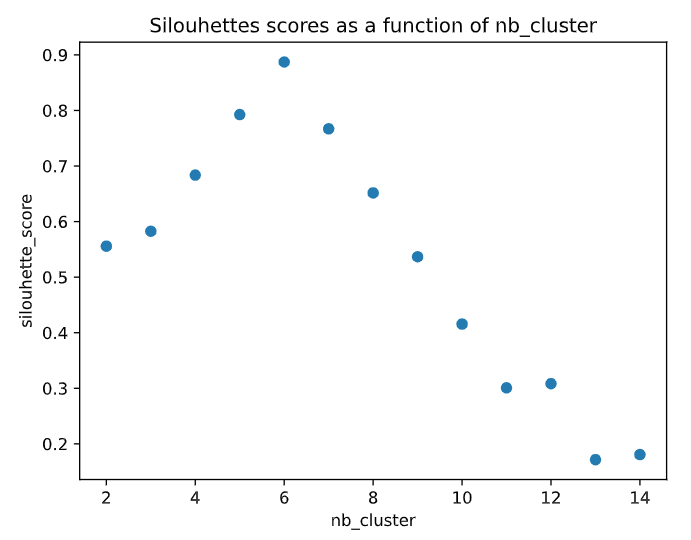
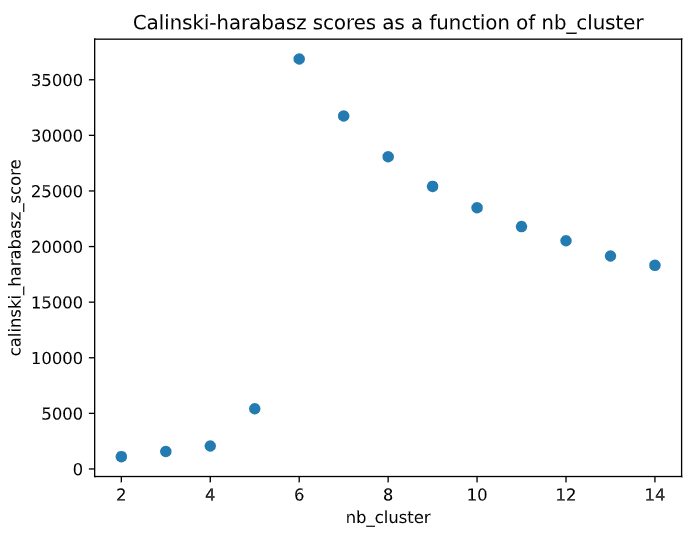
I finally decided to use Calinsky Harabasz score as last heuristic to try a way to locally determine if position is good instead of based to specific point.

Here are results I got:

**KMean:**

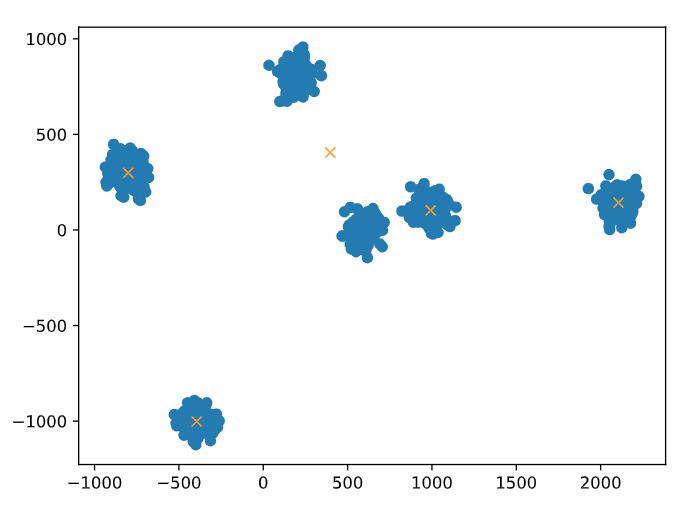
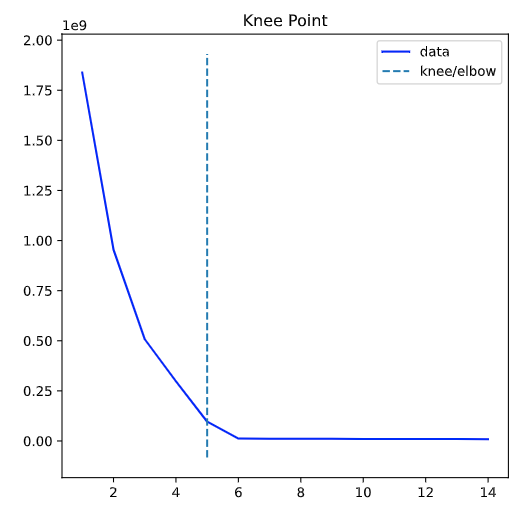
**Silhouette Score: Calinsky Harabasz Score:**

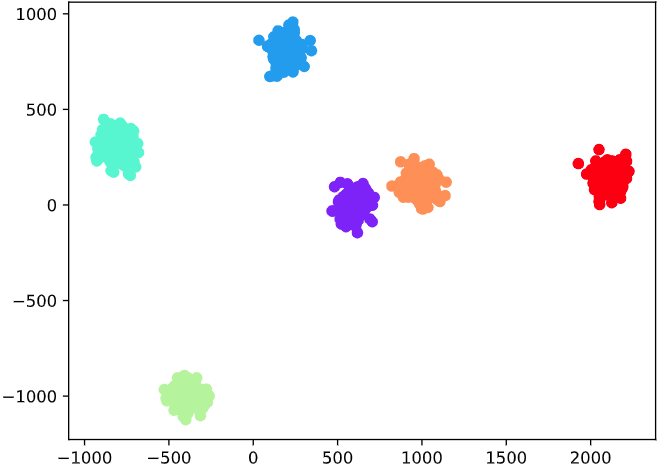
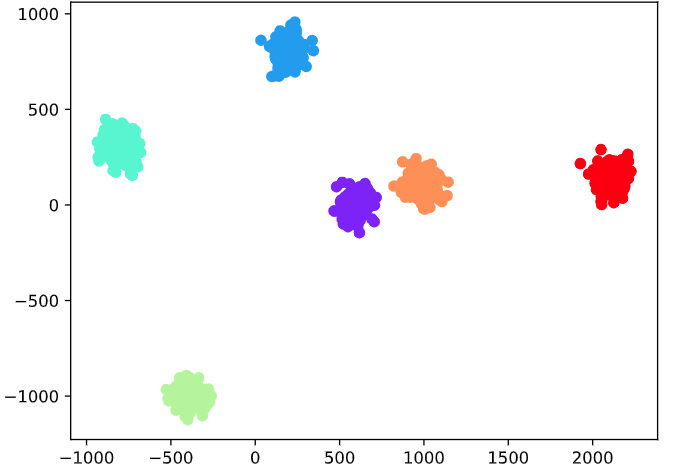
We can easily see that both this heuristics worked well with with KMean clustering method. What we can also observe is that Silhouette score gives better results for low numbers of clusters but Calinsky Harabasz score gives better results with high amount of clusters.

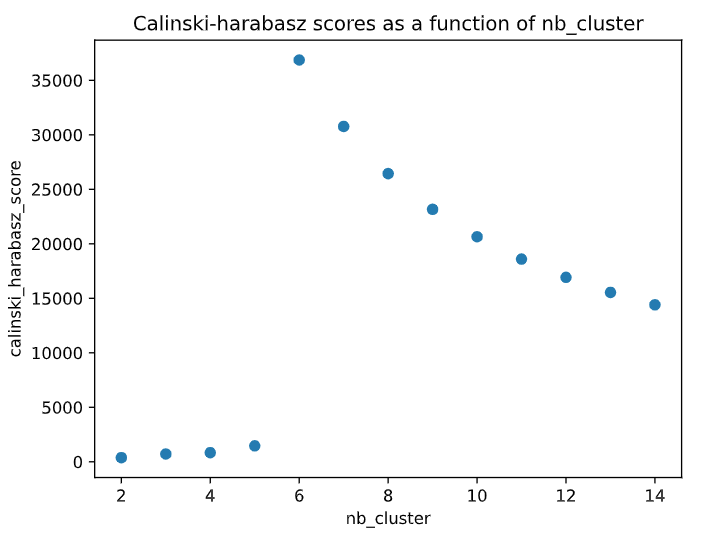
More of that, we can see that in next schematic that Kneed was not working that well on this dataset and heuristics used are more relevant here.

**Agglomerative Clustering:**

**Silhouette Score: Calinsky Harabasz Score:**

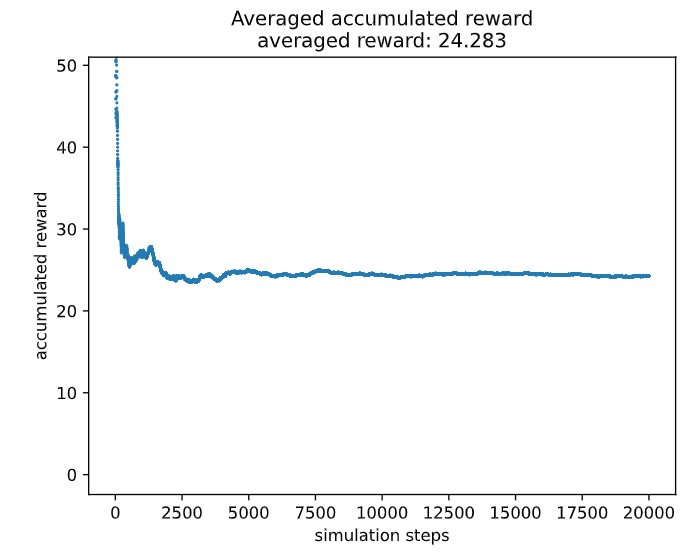
We got almost same results than with KMean with Agglomerative Clustering method but here Silouhette score seems to give better value as amount of clusters increase also.

I prefer on my side the Agglomerative Clustering Method with Calinsky Harabasz Score as heuristic. I find it easier to read and understanding how it’s working and result on score is clearer.

*Part 4: Exploitation / exploration compromise*

On this exercise I decided to use simple approach to start. I wanted my agent to look on sides if a reward was higher than in his current position and to move on it if it was so case. If it was not, I wanted him to pick random action to explore world and be able to check if rewards were on other unexplored cases.

After implementing that I tested once to see how much I had to close the gap, but average result was already higher than 20 (around 23/24). I so decided to move on next exercise and come back on it to improve it if I had more time.



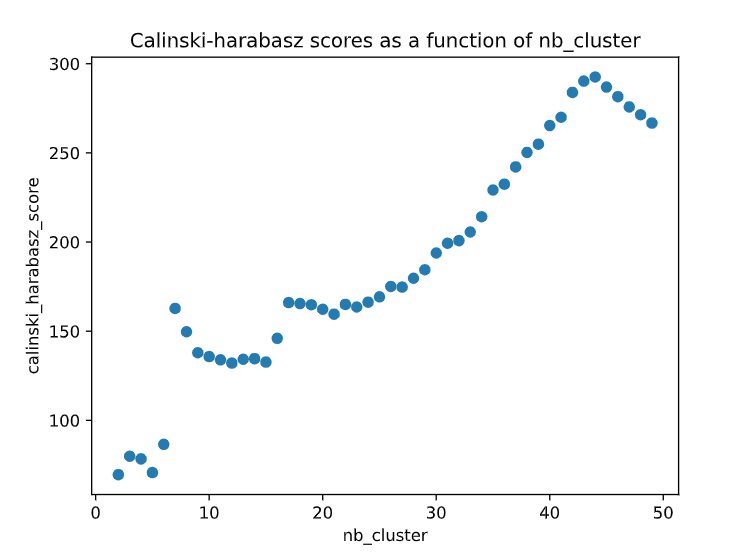
*Part 5 : Application of unsupervised learning*

For this exercise, I tried a lot of different datasets before to find the one I finally worked on.

**FIRST DATASETS EXPERIMENTS:**

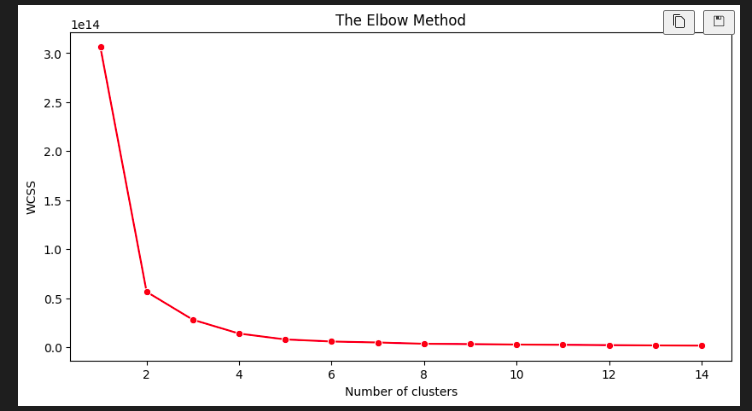
I wanted to do clustering, so I searched in links provided about interesting datasets for experimenting some clustering algorithms. I performed it and spent a lot of time on it before understanding these datas were far too complicated for a few experienced developer like me.

After some experimentations I got some interesting scores for my clustering but I found out that it gaves me as many clusters as my number of people taking part in the survey :



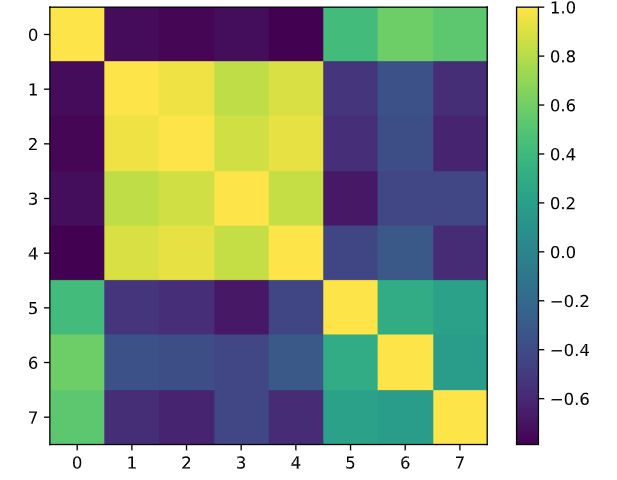
I was really frustrated about this, and I couldn’t get how to manage as data was hard to visualize.

I, after, chose some cars characteristics to try to cluster them and fin some vehicle types thanks to the data. But after many tries, I also couldn’t get a result interesting enough to do a good clustering.

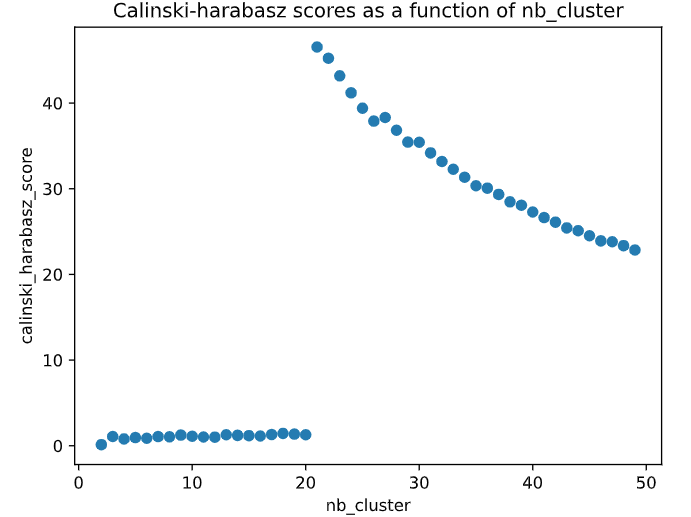


I always got 3 types of vehicles instead of around 10 I should have found for getting a result looking like vehicle types. I so chosen to try dataset with much more works on it to be able to be helped by already existing projects.

On another car dataset I had also redundancy issues and was not sure about best way to handle it so preferred to find another way. After some time, I think now it could have been a really interesting one if I just chose one of these redundant characteristics and try PCA on dataset:



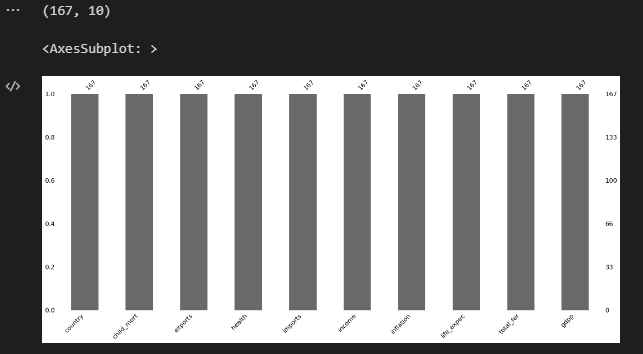
*Correlation matrix*



I also tried some on Spotify dataset but couldn’t find an interesting point of view of clustering so quickly chose to fin another very simple dataset.

**FINAL DATASET EXPERIMENTS:**

I first started to understand data I got. So I printed the shape and the amount of data in each column to check of utility to remove lines with missing data.



We can here observe that these parameters are giving us some information about important values to detect if a country is well developed or may need help to develop properly and catch biggest ones.

I then printed some data to see the shape and check if there was a magnitude difference between some parameters and if some parameters were categorical. I for example printed means and std for parameters to detect this.

Une image contenant texte

Description générée automatiquement

I then checked for correlation matrix to spot some redundancy in parameters.

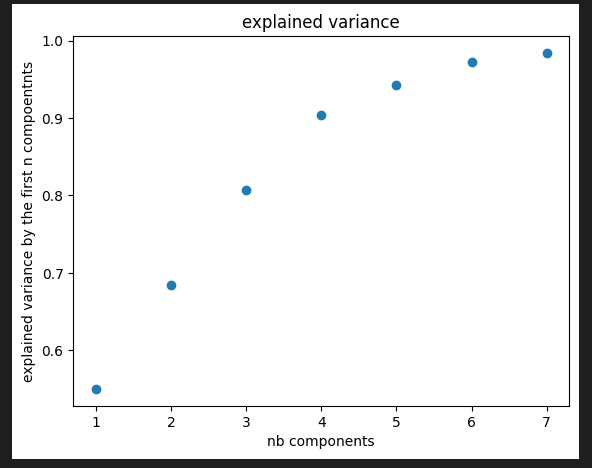
Une image contenant table

Description générée automatiquement

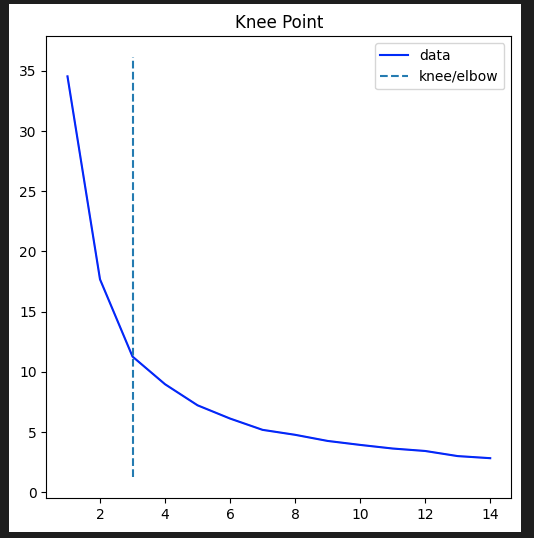
Thanks to correlation matrix we can detect that lot of parameters are relatively correlated (export – import – income…)

This help us to figure out that PCA should be useful to create a lower parametric dataset based on this one and maybe be able to plot this data.

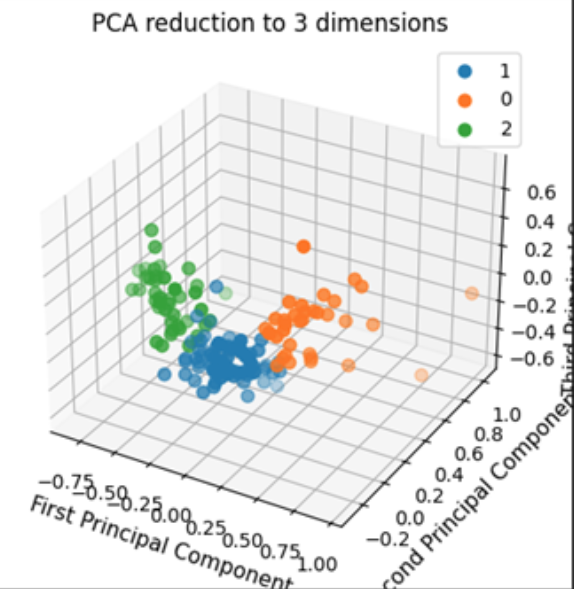
So after dropping country name as it won’t be useful for our analysis, I performed PCA with explained variance calculus to get best reduction possible to keep more than 83% of variance in dataset:



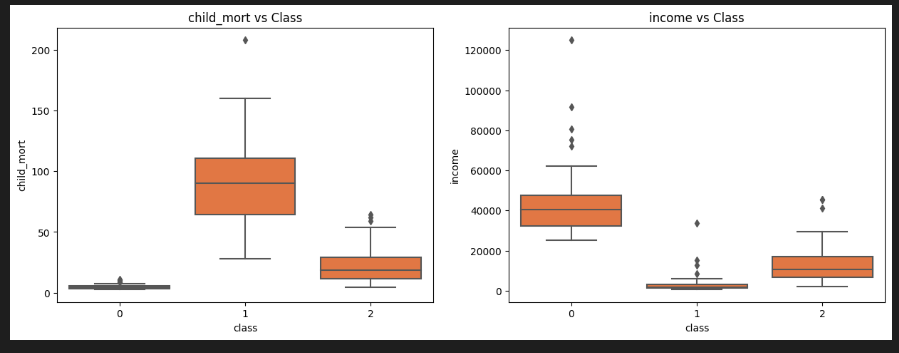
Based on this data, I performed KMean with Kneedle algorithm to get the most interesting number of clusters:



I finally performed KMeans algorithm with 3 clusters and plot it in 3 dimensions to get to this result:



I then transferred these labels obtained thanks to clustering to original dataset and plot some information about values in child mortality and income to detect which cluster is more sensible to need help or not.



I finally labelled it and saved it in result file.

Une image contenant table

Description générée automatiquement