**This document described the JSON files created for visualizing clusters in interactive manner.**

**Data structure description**

data = {}

data['feature\_names'] = row

data['X'] = np.around(x,2).tolist()

data['Y'] = y\_full.tolist()

data['clusters'] = y\_kmeans.tolist()

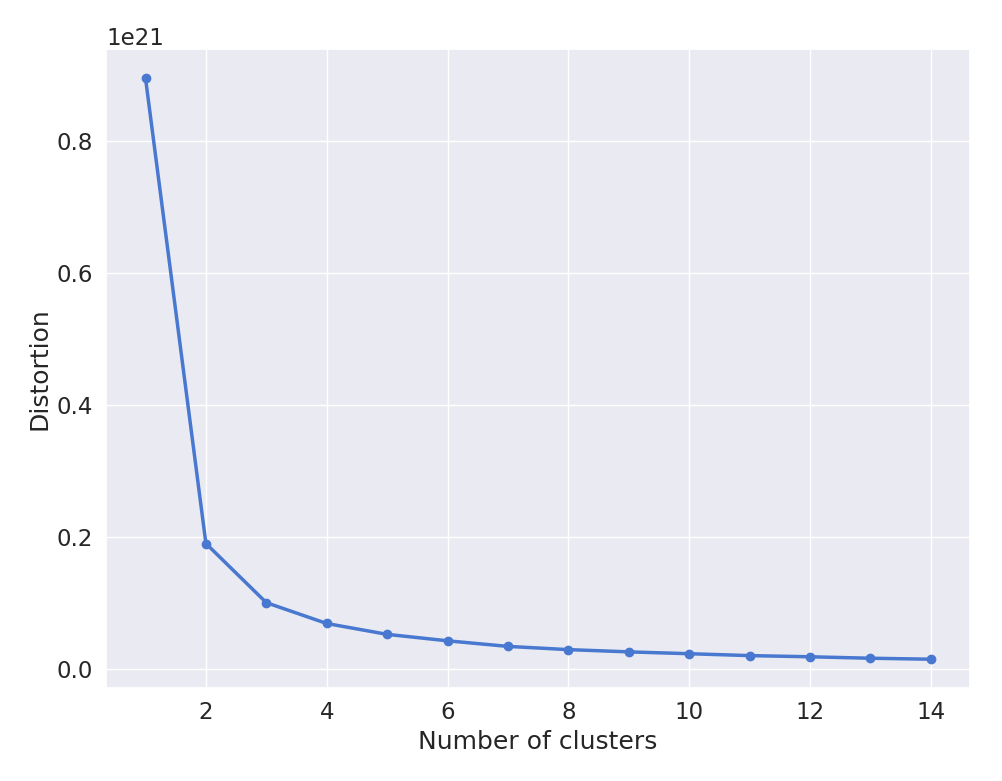
data['cert\_values'] = dataset[:,0].astype(np.int32).tolist()

* feature\_names is an array of strings containing the name of each feature used.
* X is a 2D matrix where rows are instances of the whole dataset and columns are the 21 features used in the same order of feature\_names.
* Y is a 1D list with class for each instance (row of X) 0 = non failed, 1 = failed
* clusters is a 1D list with cluster number associated with each instance (row of X). E.g. for 2 clusters dataset, only clusters 0 and 1 will be available; for 3 clusters there will be cluster 0, 1 and 2.
* cert\_values is a 1D list of integers of CERT unique values associated to each instance (row of X) of the dataset.

**The selection of the number of the clusters**

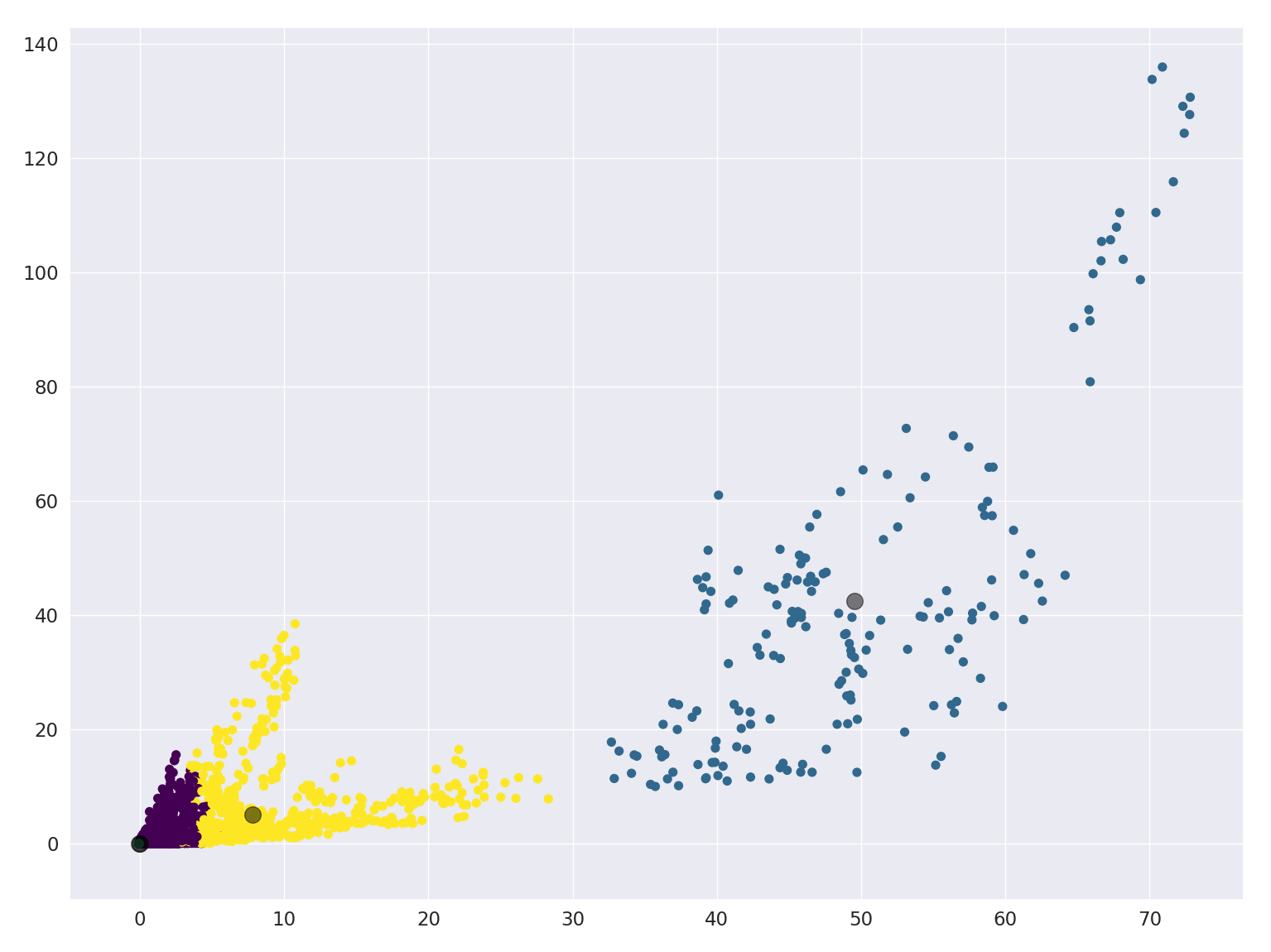
I noticed that DBSCAN took too much time to compute giving, at the end, a memory error (despite my 32GB ram). I then switched to K-Means.

I have defined the optimal number of clusters for the dataset looking at the elbow curve:



The best number of clusters is between 2 and 4.

Follows an example of the clustering using the first two dimensions of the dataset and 4 clusters:



As it’s possible to note it’s almost impossible to notice the 3th cluster which is hidden below the first (violet).

Probably using other features the 4th cluster is more visible.