DBSCAN vs KMeans

March 25, 2020

1 Time efficiency:

We tested the two algorithms on two datasets of different sizes on the platform (https://www.kaggle.com):

The first dataset has 150 lines and a size of 4.99KB

The second dataset has 4800 lines and a size of 158KB

For 10 tests we obtained the following results:

Time results

ALGORITHM	DS_DIMENSION	MIN	MAX	MEAN	MEDIAN	SD
DBSCAN	4.99KB	0.00398s	0.00498s	0.00410s	0.00402s	0.00030s
KMeans	4.99KB	0.01993s	0.02894s	0.02511s	0.02546s	0.00265s
DBSCAN	158KB	0.08462s	0.11622s	0.09987s	0.1s	0.00764s
KMeans	158KB	0.03993s	0.04775s	0.04275s	0.04251s	0.00235s

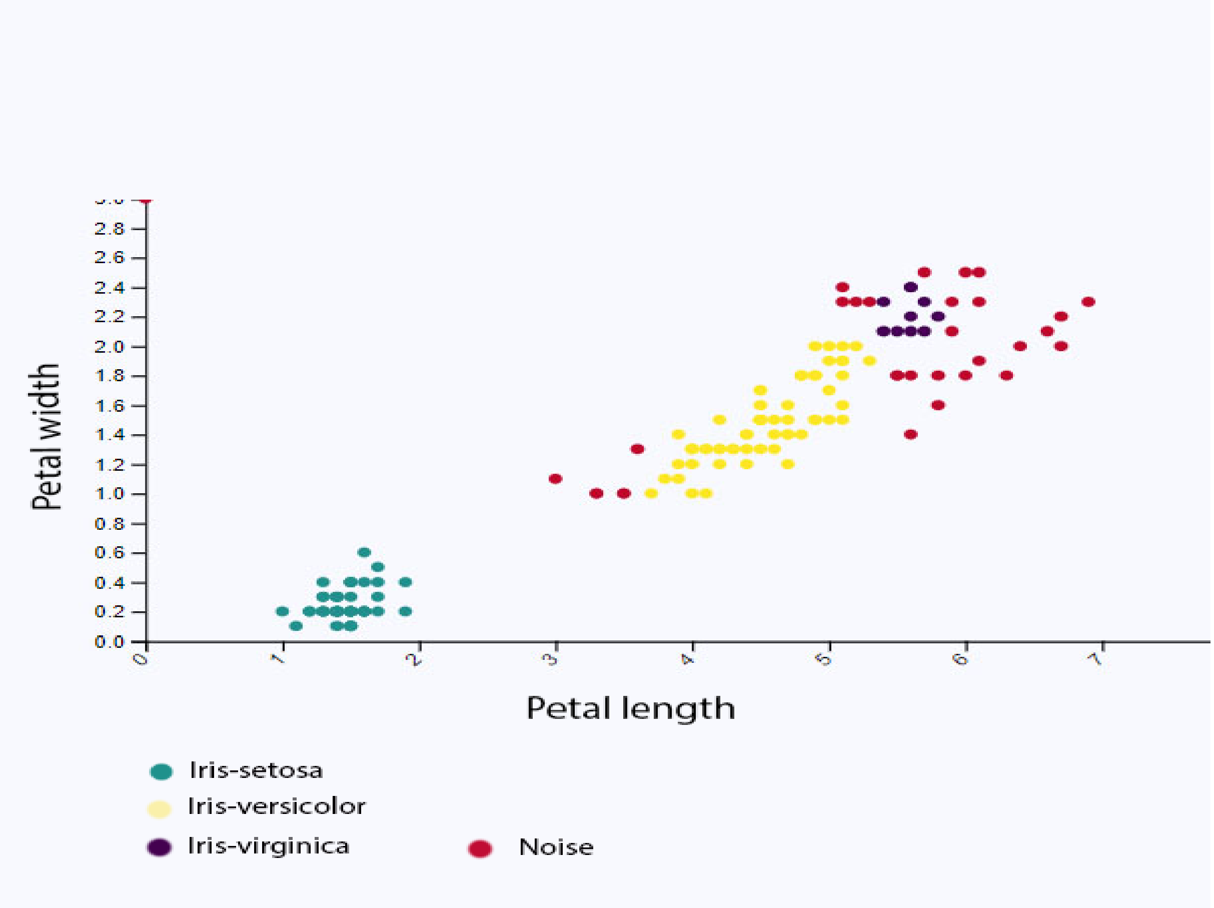
In conclusion on small datasets KMeans is faster than DBSCAN but on large datasets KMeans is much faster than DBSCAN this happens because DBSCAN creates its own clusters which lasts longer than you already know their number, as in the case of KMeans.

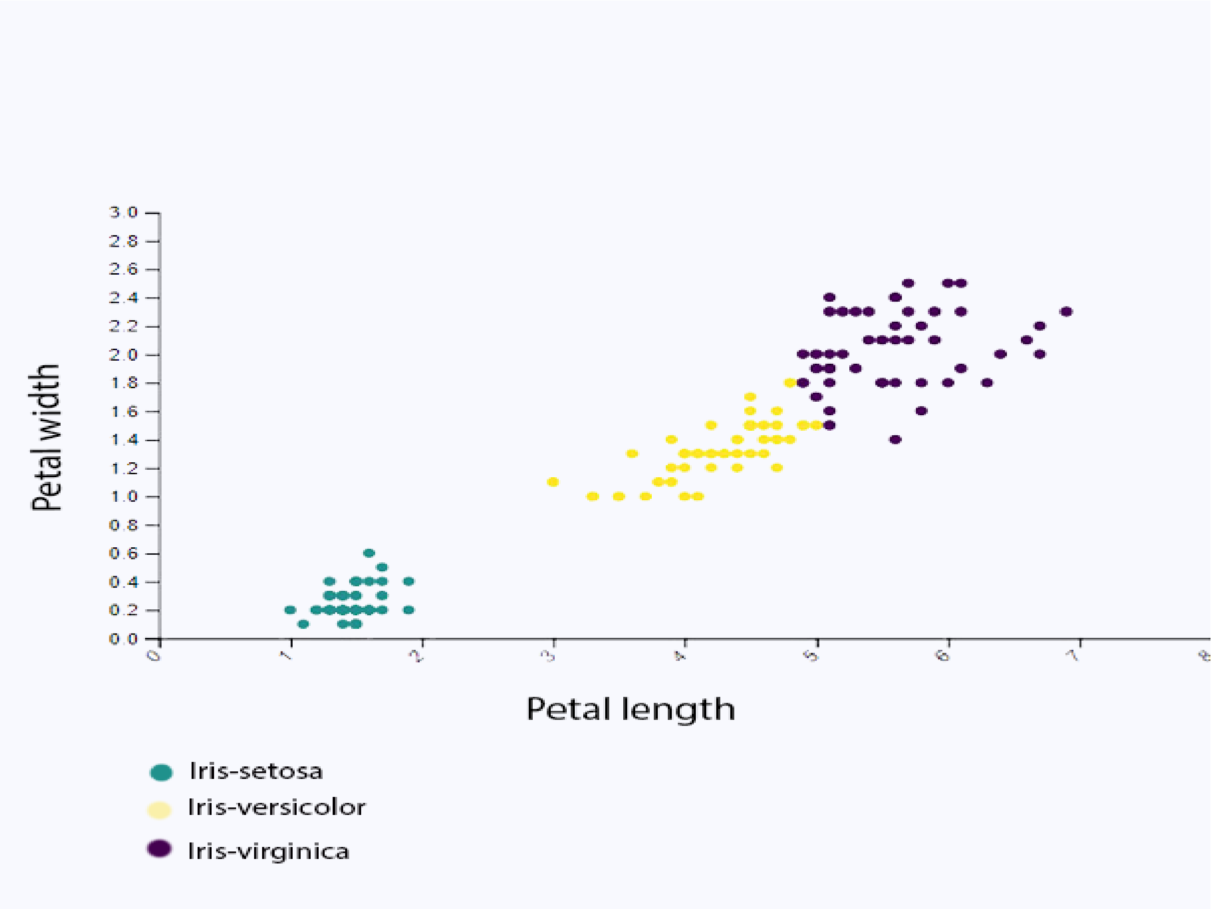
A vulnerability of the k -means is the number of clusters too large, if a large number of clusters is specified, it lasts a long time until the centroid is properly set.

2 Advantages/Disadvantages of algorithms

A disadvantage of DBSCAN is that it does not work well on clusters with different densities:

In the following images we have given two examples of clustering. The first example for DBSCAN and the second for KMeans





From the two images it can be seen that due to the different densities the DBSCAN has a lot of noise, while KMeans did better in dividing in clusters.

disadvantage of KMeans is that we should know the clusters beforehand while DBSCAN does not need to know how many clusters are but we must be more careful with the parameters (eps amd min_samples)