Esra Eryılmaz
171044046

CSE 454 – Data Mining (Fall 2021)
Assignment #2
Report

Prepare a report that presents the results for the questions mentioned below using two datasets, where DS1 has two dimensions and DS2 has at least 20 dimensions. You have to find the datasets yourself.

DS1 ⟶ has two dimensions

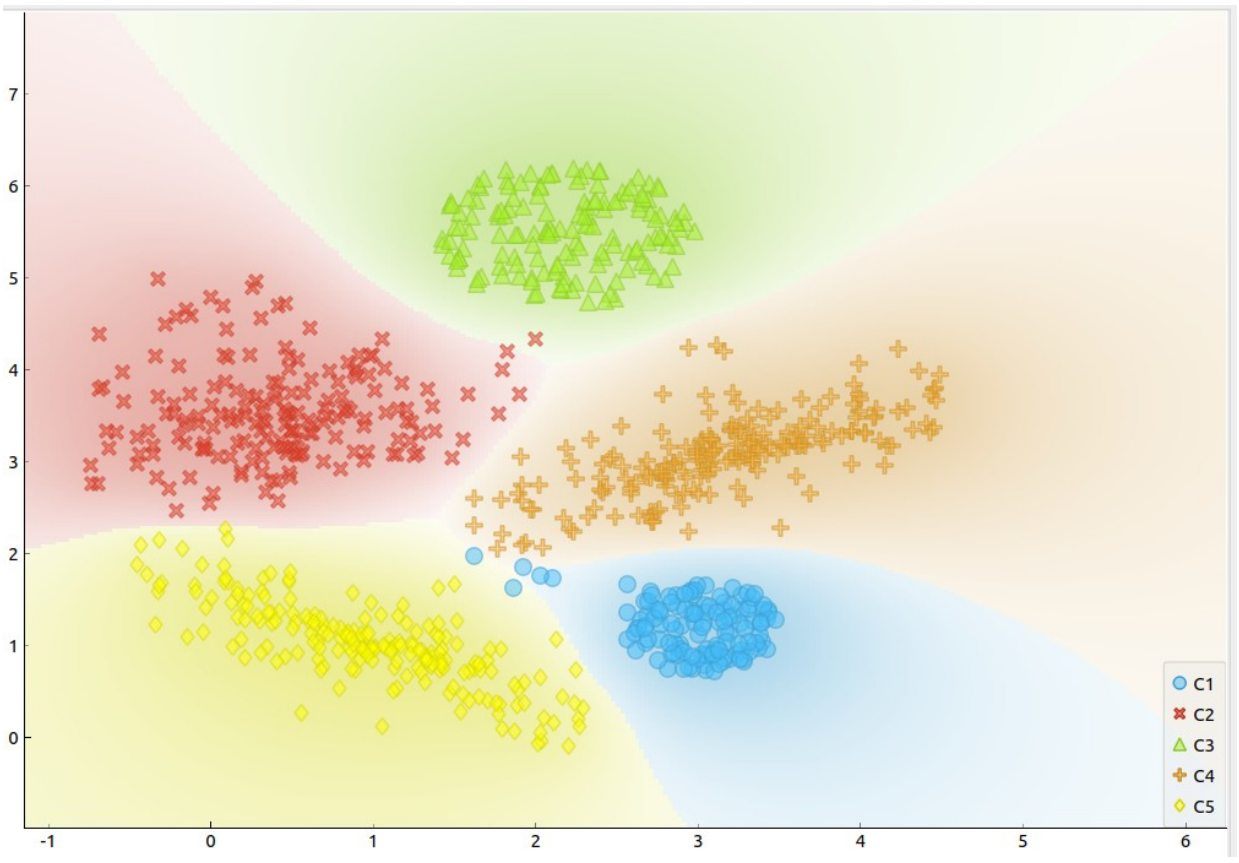
DS2 ⟶ has twenty five dimensions

(I used Orange and Weka data mining tools for finding results and evaluate results myself.)

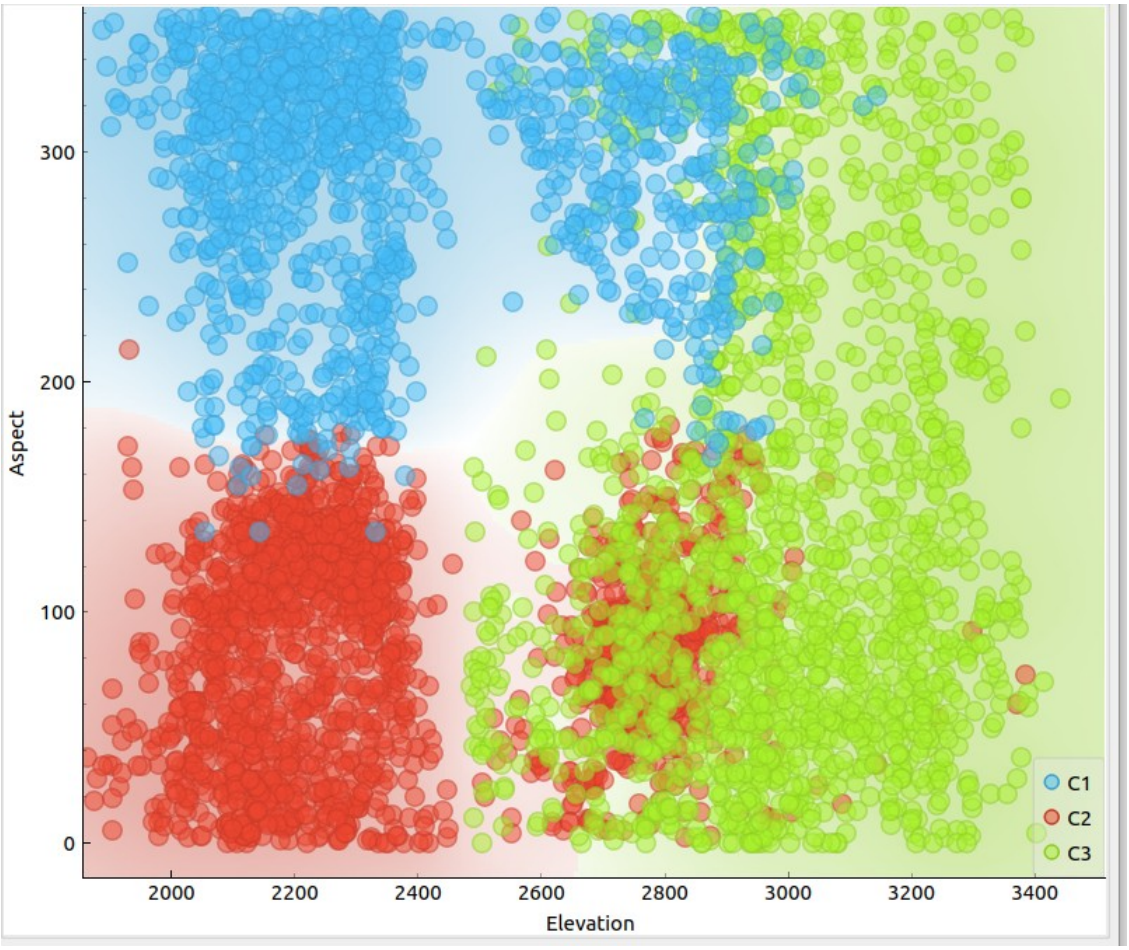
1) Find clusters using Frequent Pattern Growth, k-means, DB scan and Chameleon clustering techniques. You may use data mining tools to find clusters.

K-means

2D

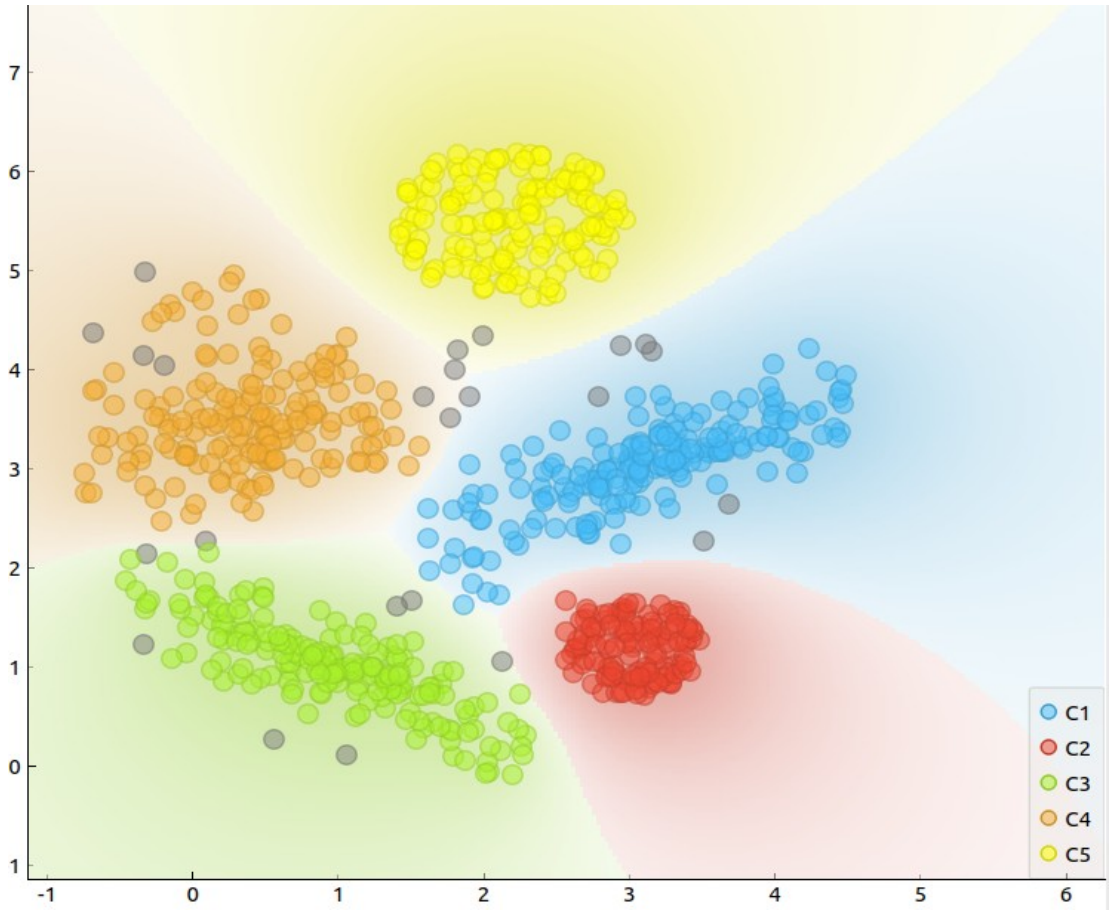


25D

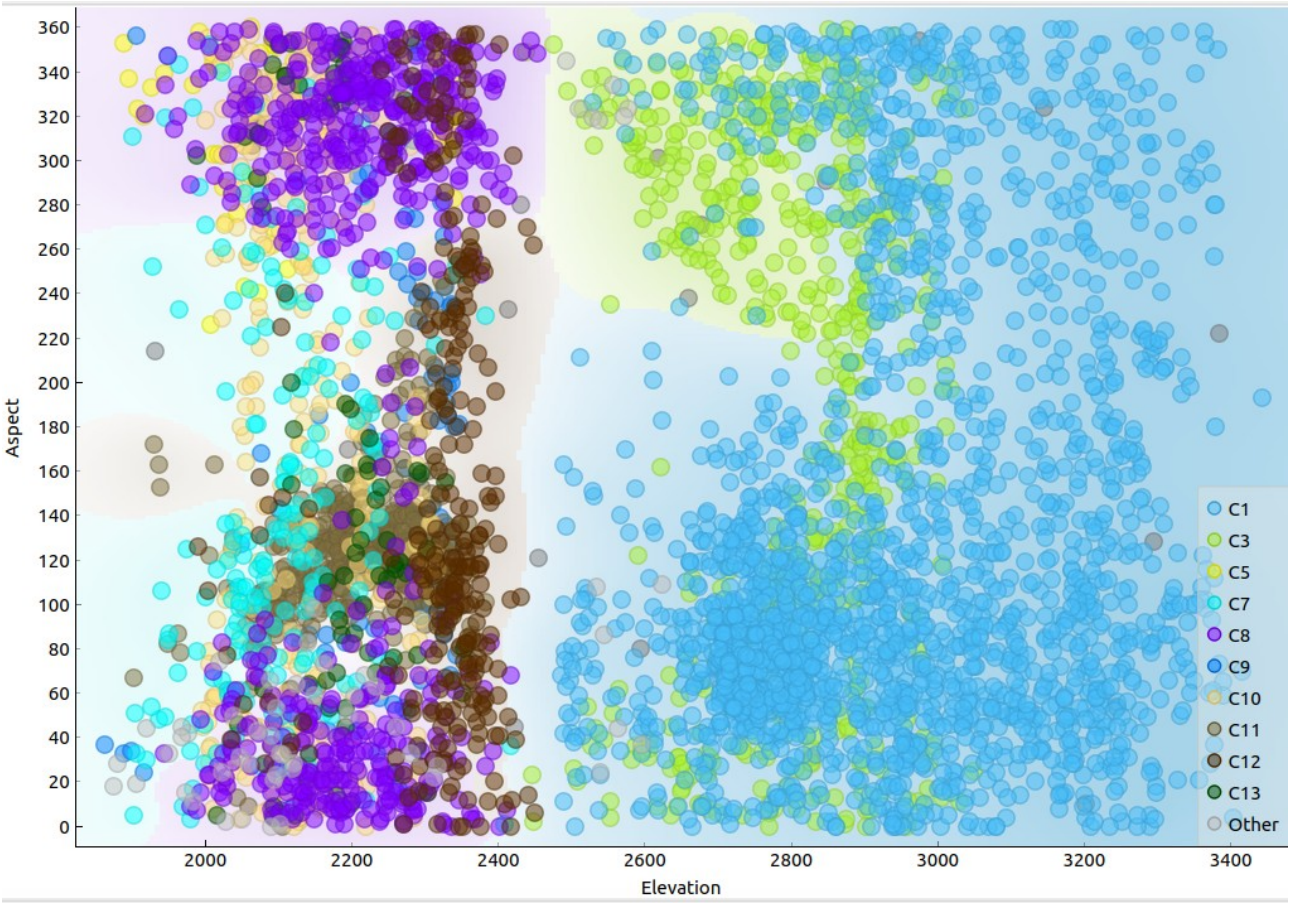


DBSCAN

2D

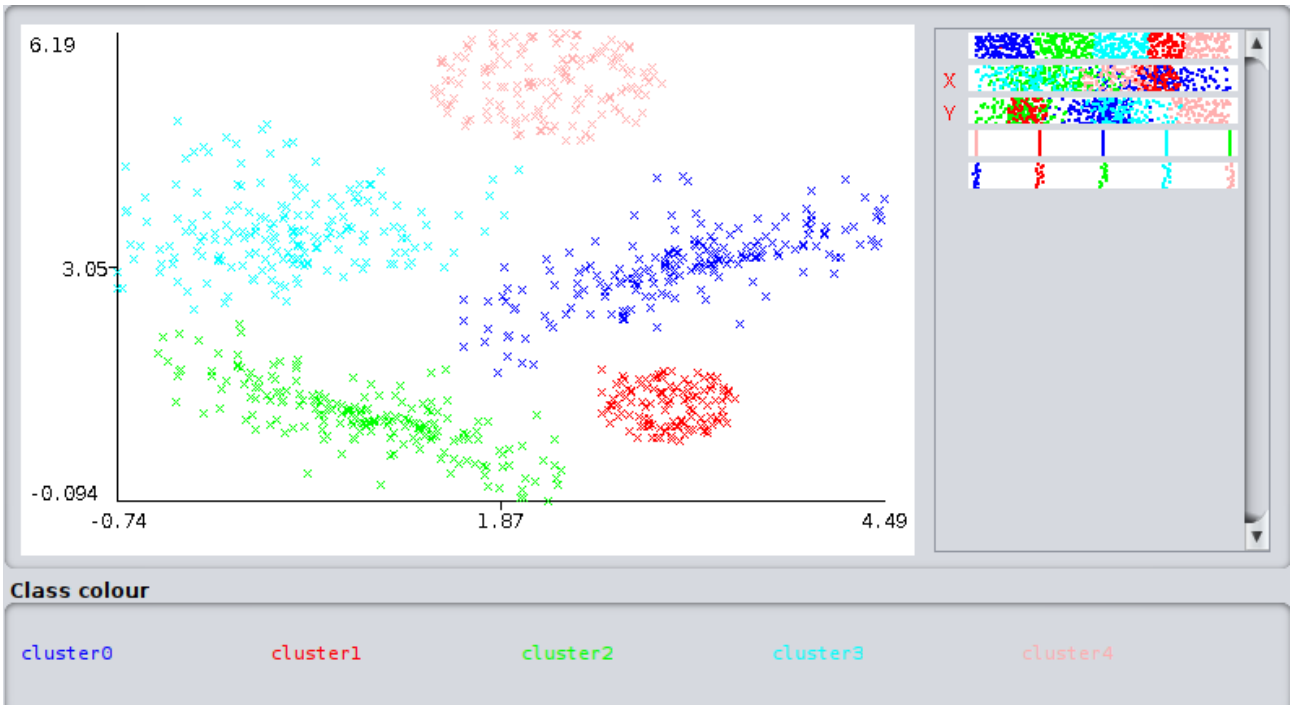


25D

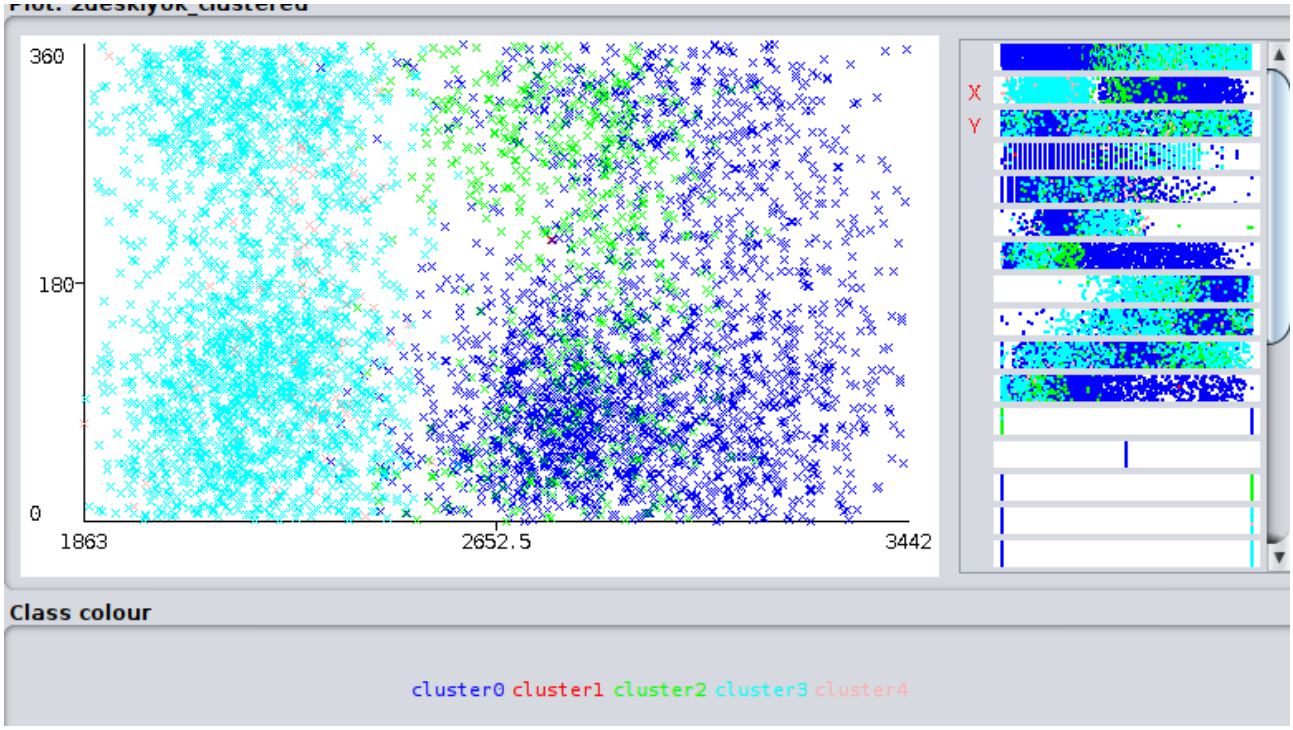


Chameleon

2D



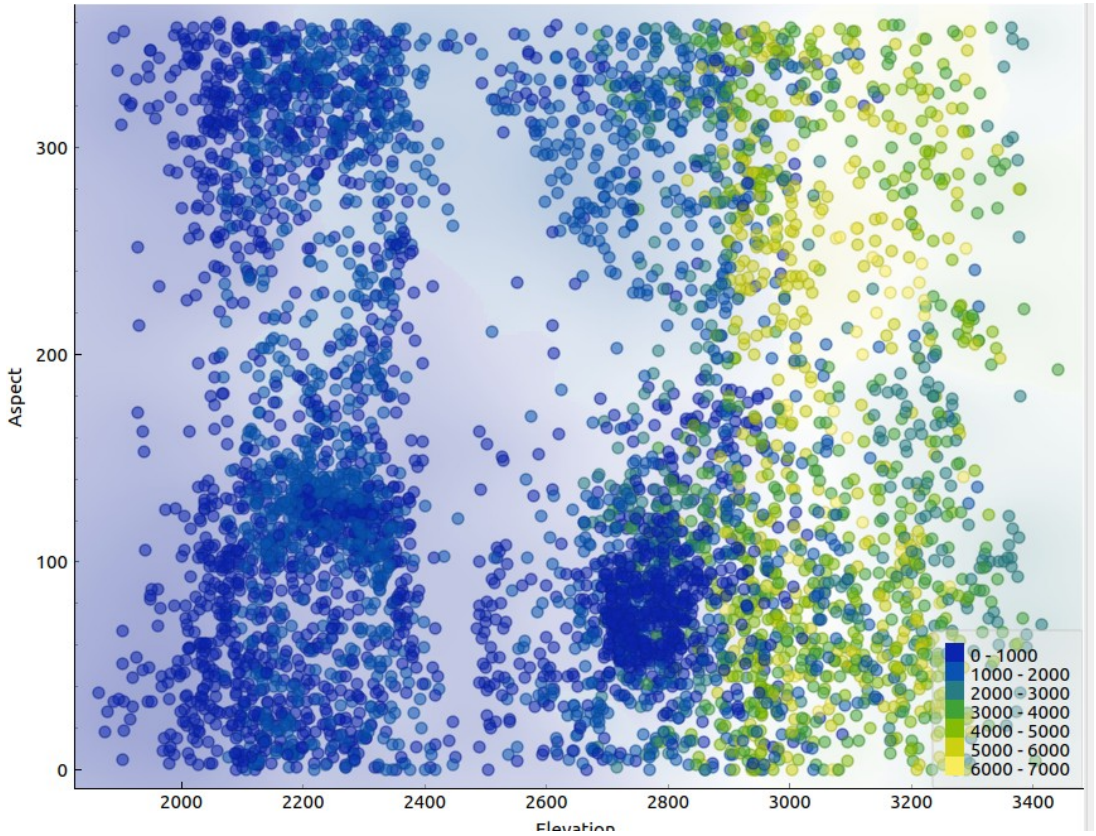
25D



FPGrowth

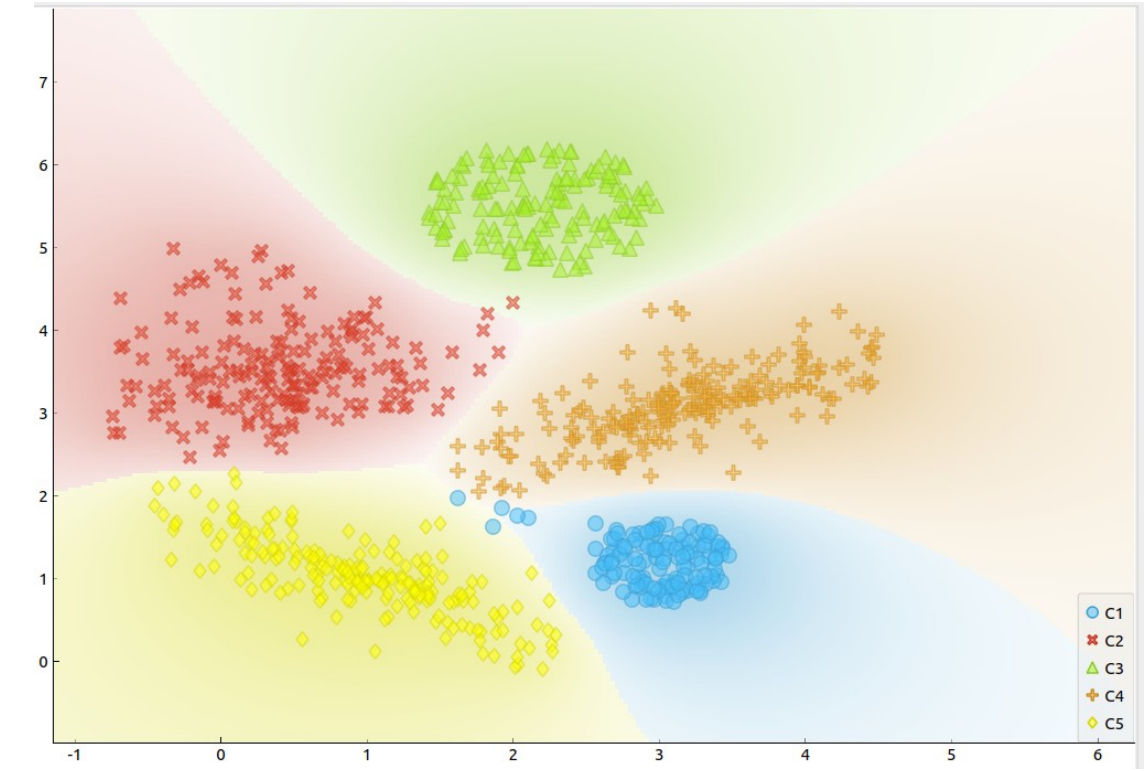
2D

25D

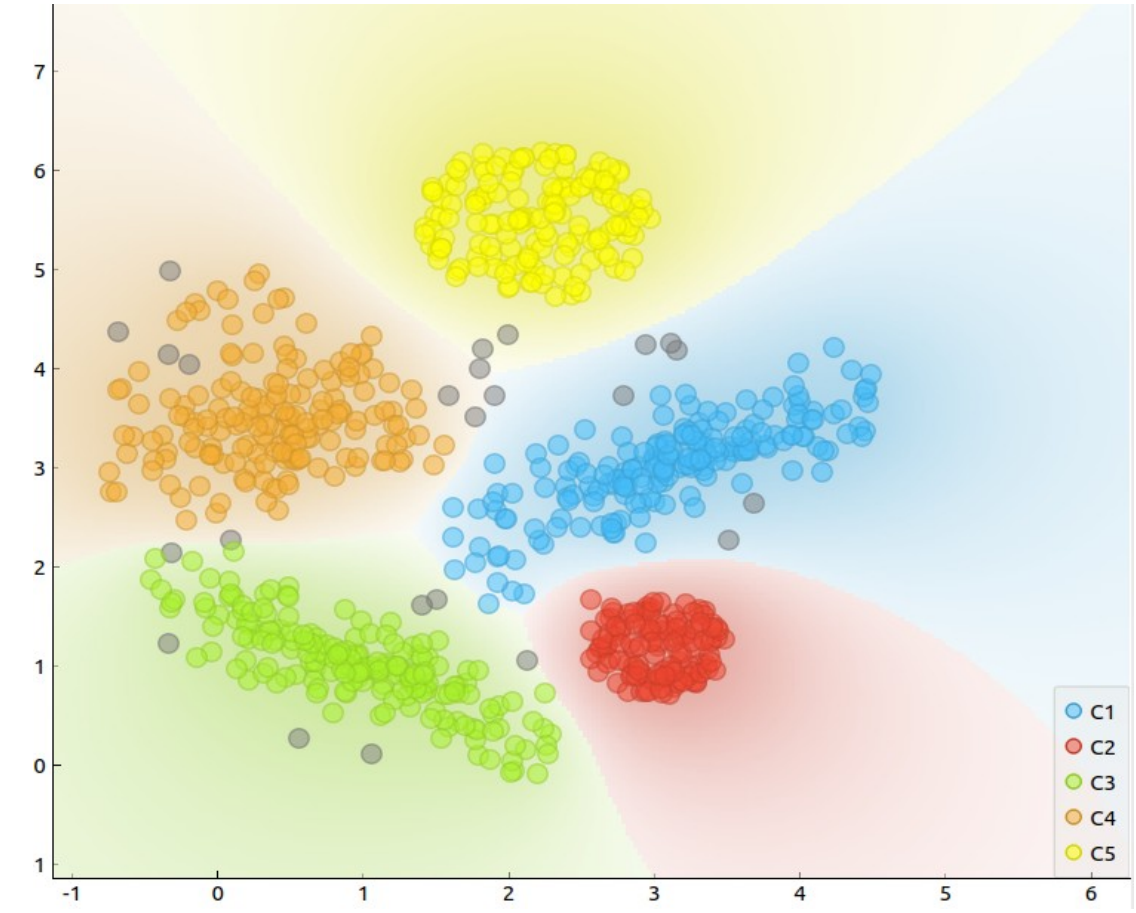


2) Present the clusters of DS1 for each technique using graphics.

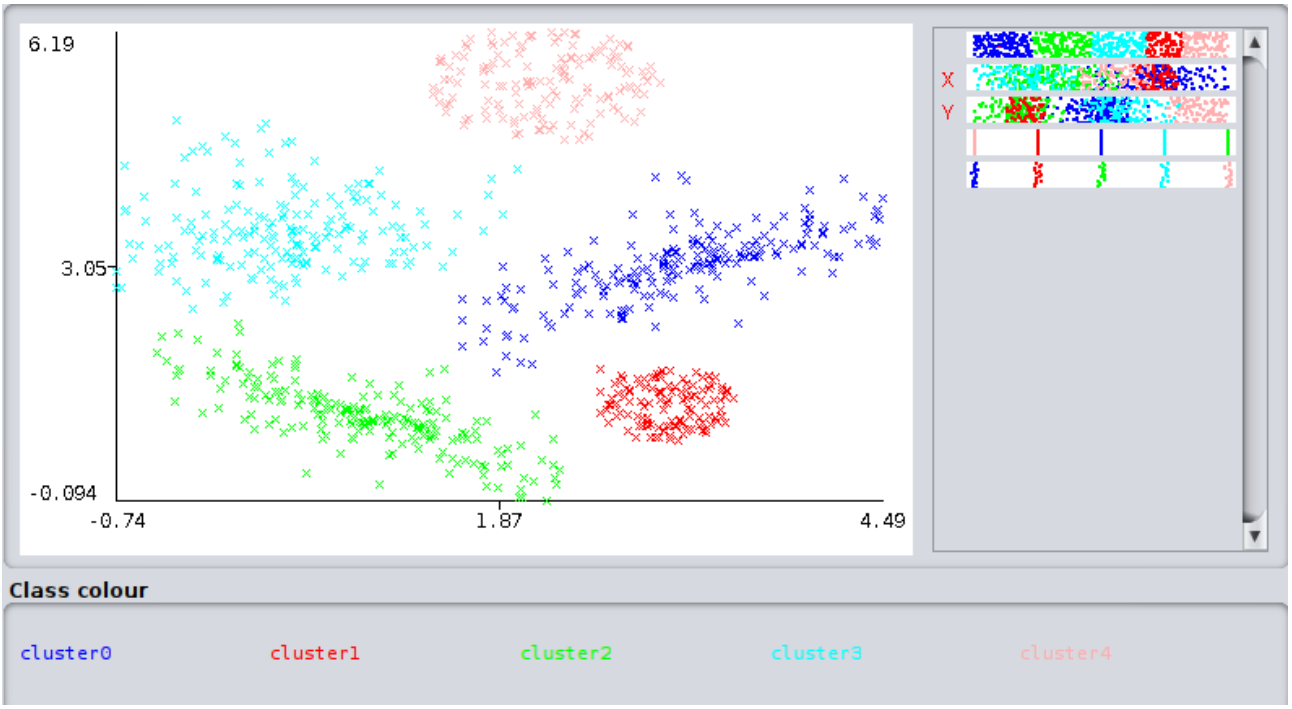
K-means :



DBSCAN :



Chameleon :



3) Calculate silhouette coefficient for each clustering technique. Compare and interpret the silhouette score with the extracted clusters.

$$s(o) = \frac{b(o) - a(o)}{\max\{a(o), b(o)\}}$$

where,

- ***s(o)*** is the silhouette coefficient of the data point **o**

- ***a(o)*** is the *average distance* between **o** and all the other data points in the cluster to which **o** belongs

- ***b(o)*** is the *minimum average distance* from **o** to all clusters to which **o** does not belong

The value of the silhouette coefficient is between [-1, 1].
A score of 1 denotes the best meaning that the data point o is very compact within the cluster to which it belongs and far away from the other clusters. The worst value is -1.

K-mean silhouette coefficient values are close to the number of 1. It goes up to 0.8 . In the 2D dataset there are no negative values. In 25D dataset there are some negative values up to -0.6.

DBSCAN silhouette coefficient values have more positive values. It goes to 0.6 . Rather than K-means in 2D dataset there are a few negative values. In 25D dataset it is similar to K-means, there are some negative values up to -0.6.

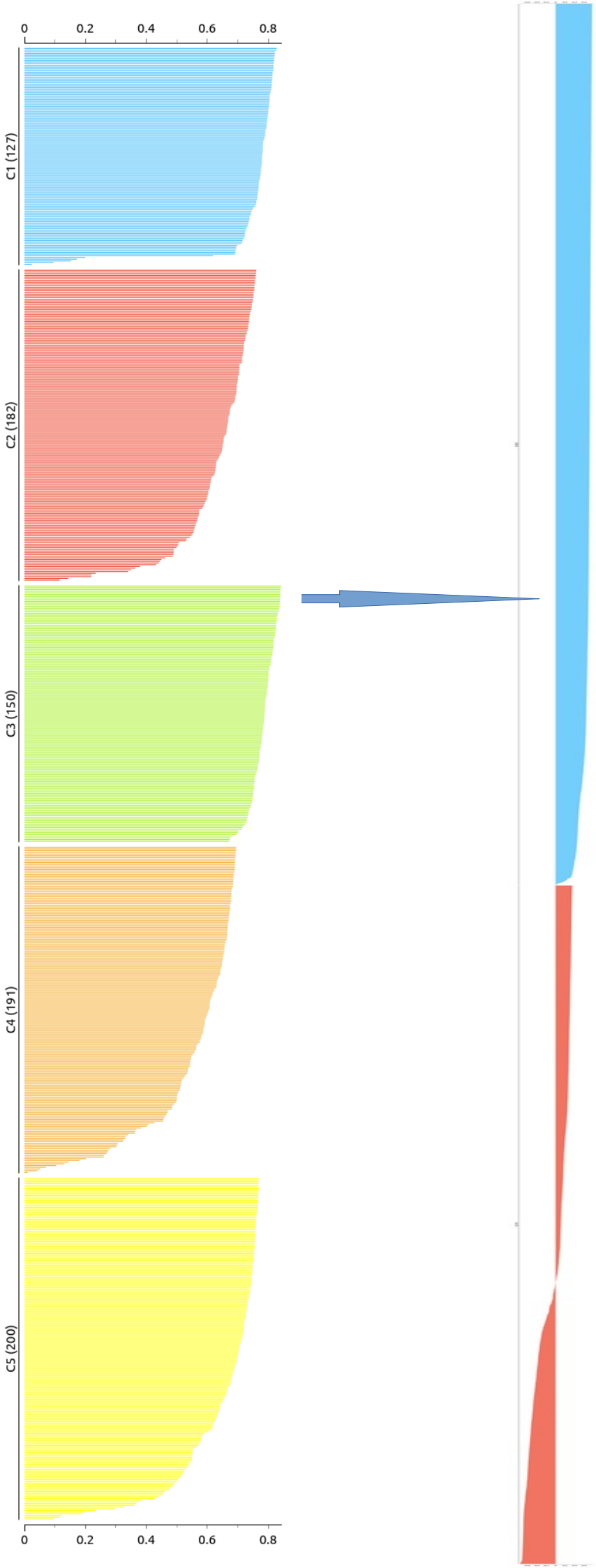
Chameleon silhouette coefficient values, in 25D dataset, there are some negative values but it less than K-means and DBSCAN.

FPGrowth silhouette coefficient values, in 25D dataset, there are some negative values but it less than K-means and DBSCAN.

K-means :

2D

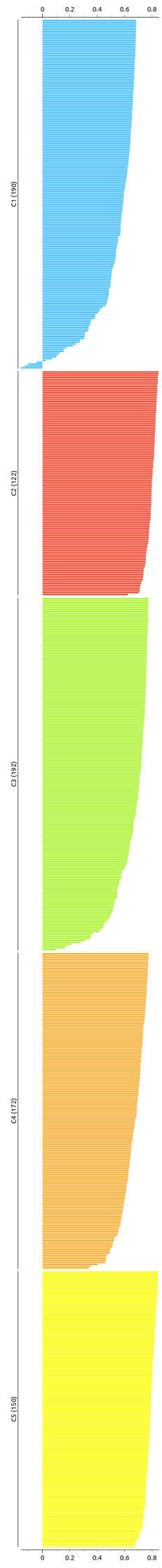
25D

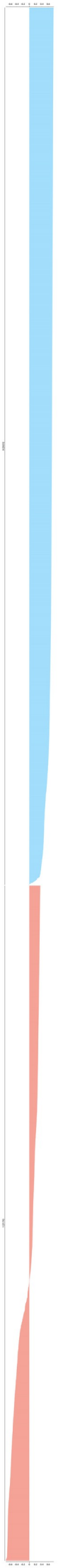


DBSCAN :

2D

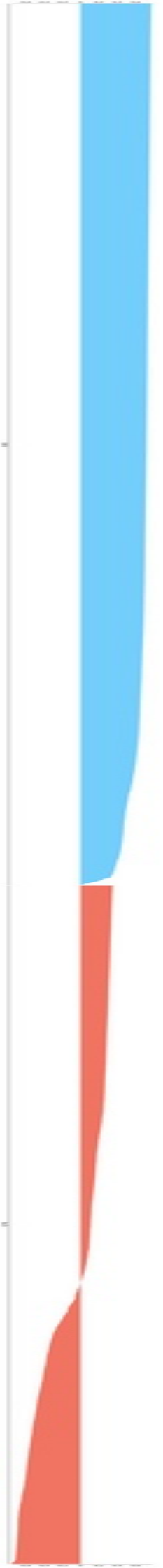
25D





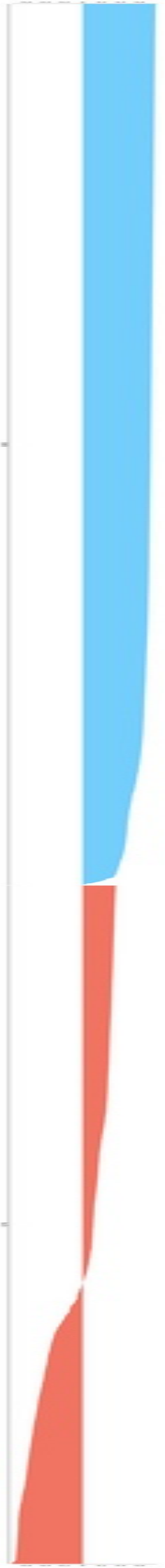
Chameleon :

25D



FPGrowth :

25D



4) Present computational time and time complexity of each clustering model.

K-means : *O(n)*

2D

```
Time taken to build model (full training data) : 0.05 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       69 (  8%)
1      113 ( 13%)
2      200 ( 24%)
3      318 ( 37%)
4      150 ( 18%)
```

25D

```
Time taken to build model (full training data) : 0.17 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       344 (  7%)
1      1357 ( 27%)
2       607 ( 12%)
3      1612 ( 32%)
4       817 ( 16%)
5       262 (  5%)
```

DBSCAN : *O(nlogn)*

2D

```
Time taken to build model (full training data) : 0.06 seconds
```

25D

```
Time taken to build model (full training data) : 3.66 seconds
```

Chameleon : $O(n^2)$

2D

```
Time taken to build model (full training data) : 0.36 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      196 ( 23%)
1      122 ( 14%)
2      200 ( 24%)
3      182 ( 21%)
4      150 ( 18%)
```

25D

```
Time taken to build model (full training data) : 51.12 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      2173 ( 43%)
1         1 (  0%)
2       606 ( 12%)
3      2116 ( 42%)
4       103 (  2%)
```

FPGrowth : $O(n^2)$ *(worst time complexity)*

The complexity depends on searching of paths in FP tree for each element of the header table, which depends on the depth of the tree.
So it has good complexity for my dataset.

5) Which clustering technique is more suitable for your dataset? Write a discussion about it using the results mentioned above and characteristics of the clusters and the dataset.

In the results of the 25D datasets, visualizations are not looking good. Because it has 25 dimensions so it is nested, it looks complicated. But 2D visualizations looking good.

K-means is a partitioning clustering algorithm. For K-means, I gave the cluster numbers and it calculates. I sometimes change my numbers for good results. Actually it gives good results but according to my cluster number input. I think without my input it is not usefull algorithm. And K-means time complexity is good both 2D and 25D.

DBSCAN is a density based clustering algorithm. For DBSCAN, it gave some outliers rather than K-means. I gave maximum distance for the epsilon neighborhood and the number of points which are required to form a region. It changes according to this inputs. For the time complexity it is not bad. 2D is good, but 25D is running a little bit more.

Chameleon is a hierarchical clustering algorithm. For Chameleon, it gaves good results too. Clusters are not bad but time complexity is worst. In 25D dataset, it took almost 1 minute. It uses dynamic modeling so in high dimensional datasets it took so long.

FPGrowth is a frequent pattern mining (association) algorithm. For FPGrowth I have a problem with my tools. It doesnt work for 2D, I don't know why. But I know that it has a good time complexity, the complexity depends on searching of paths in FP tree for each element of the header table, which depends on the depth of the tree.
So it has good complexity for my 25D dataset. I couldn't write exact number for time but while calculation I looked and it didn't take much time. So this algorithm is good too.

Actually all the algorithms gave me not bad clusters. I think I should compare it with complexity. In chameleon, implementation is hard, also time complexity is high. So I think chameleon is the worst algorithm for my datasets.
I think K-means is the best, because it took least time for both my datasets. But I gave cluster numbers according to that input it gaves good results.
If I don't want to give cluster numbers then I can say that DBSCAN is the best. But I gave DBSCAN some inputs too. But it doesn't effect the results so much.
If I don't want to give any inputs to the algorithm then I can say that FPGrowth is the best. Because time complexity is good for my 25D dataset.

Also if I look at the silhouette coefficient values, K-means also gave me good values, values are close to 1. In 2D dataset there aren't any negative values. Just for the 25D dataset there are some negative values. When I consider everything together I choose K-means is the best clustering algorithm for my datasets.

Additional informations :

K-means

Algorithm 1 k-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

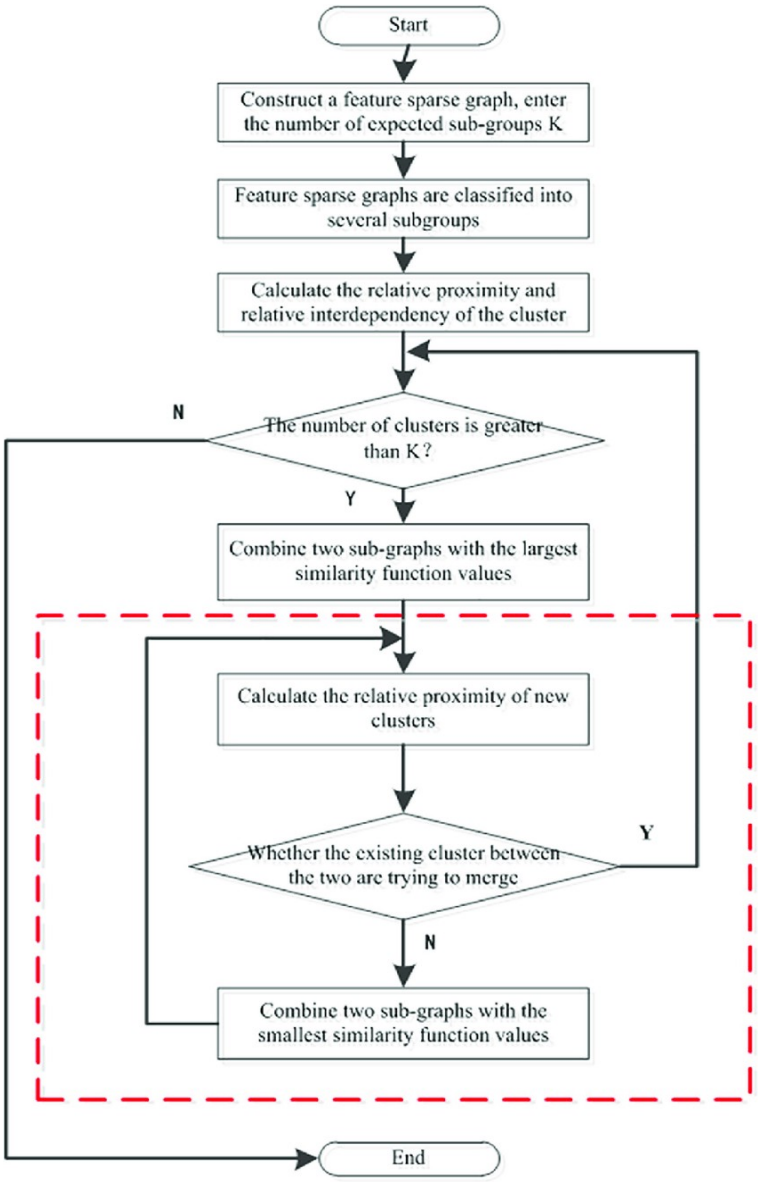
DBSCAN

```
DBSCAN(D, eps, MinPts)
   C = 0
   for each unvisited point P in dataset D
      mark P as visited
      NeighborPts = regionQuery(P, eps)
      if sizeof(NeighborPts) < MinPts
         mark P as NOISE
      else
         C = next cluster
         expandCluster(P, NeighborPts, C, eps, MinPts)

expandCluster(P, NeighborPts, C, eps, MinPts)
   add P to cluster C
   for each point P' in NeighborPts
      if P' is not visited
         mark P' as visited
         NeighborPts' = regionQuery(P', eps)
         if sizeof(NeighborPts') >= MinPts
            NeighborPts = NeighborPts joined with NeighborPts'
      if P' is not yet member of any cluster
         add P' to cluster C

regionQuery(P, eps)
   return all points within P's eps-neighborhood (including P)
```

Chameleon



FPGrowth

Input: constructed FP-tree

Output: complete set of frequent patterns

Method: Call FP-growth (FP-tree, null).

procedure FP-growth (Tree, α)

{

1) if Tree contains a single path P then
2) for each combination do generate pattern β ∪ α with support = minimum support of nodes in β.
3) Else For each header ai in the header of Tree do {
4) Generate pattern β = ai∪α with support = ai.support;
5) Construct β.s conditional pattern base and then β.s conditional FP-tree Tree β
6) If Tree β = null
7) Then call FP-growth (Tree β, β)}

}