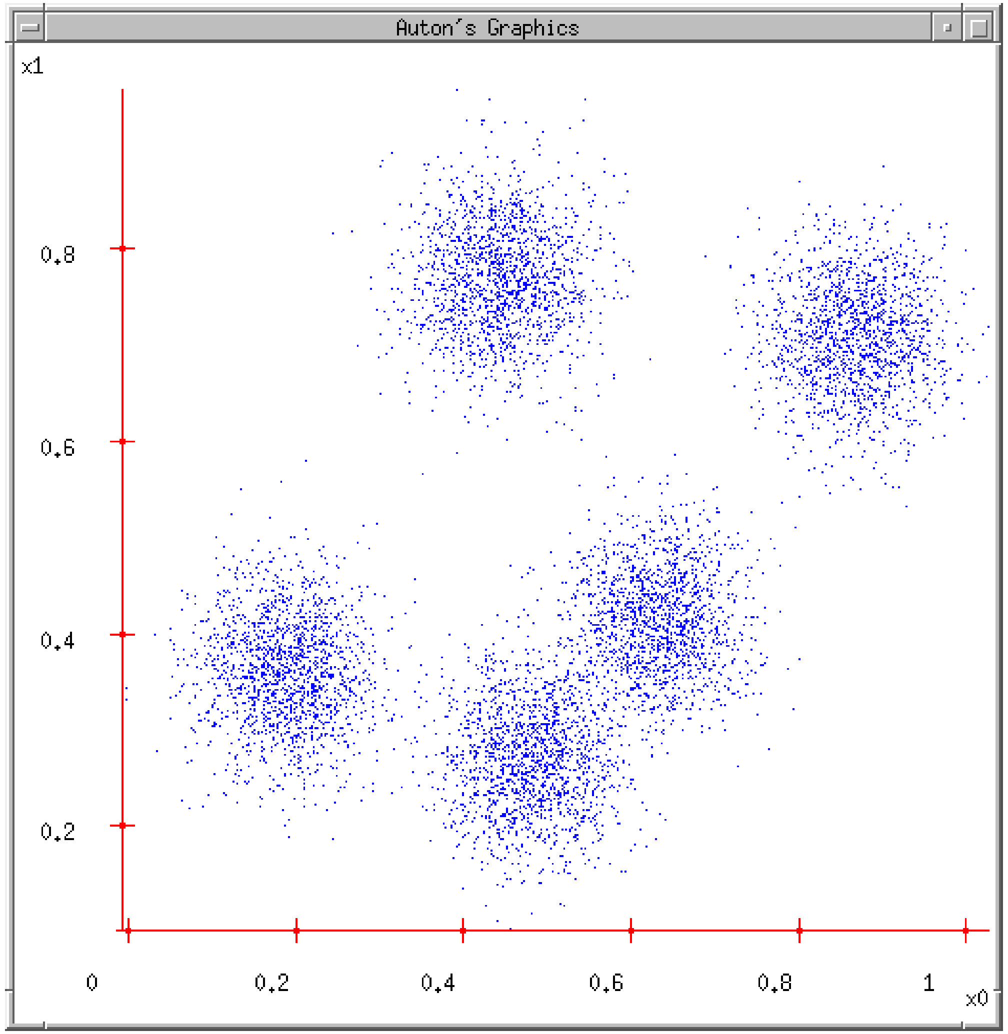
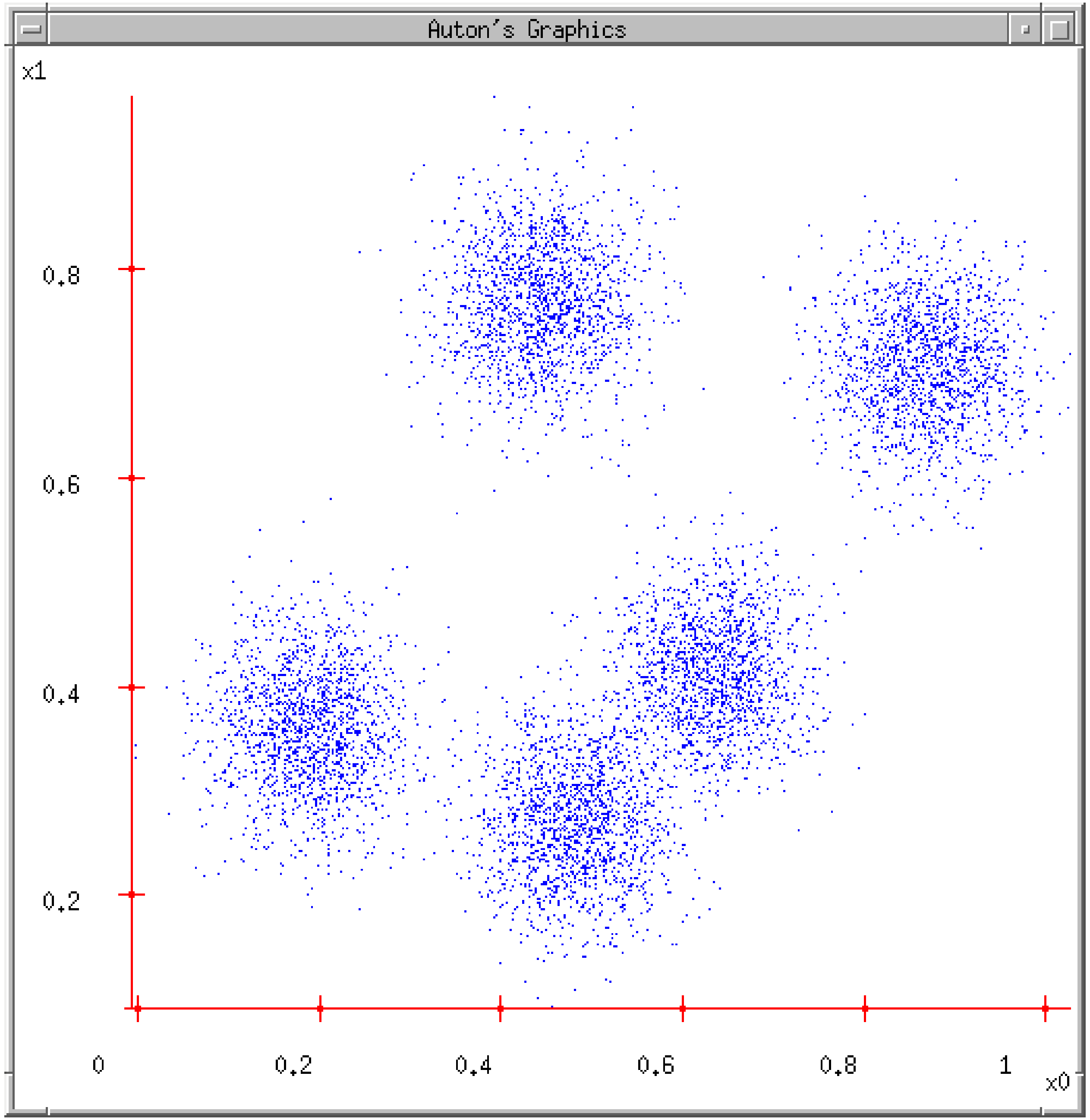
k-Means Clustering

Clustering Data



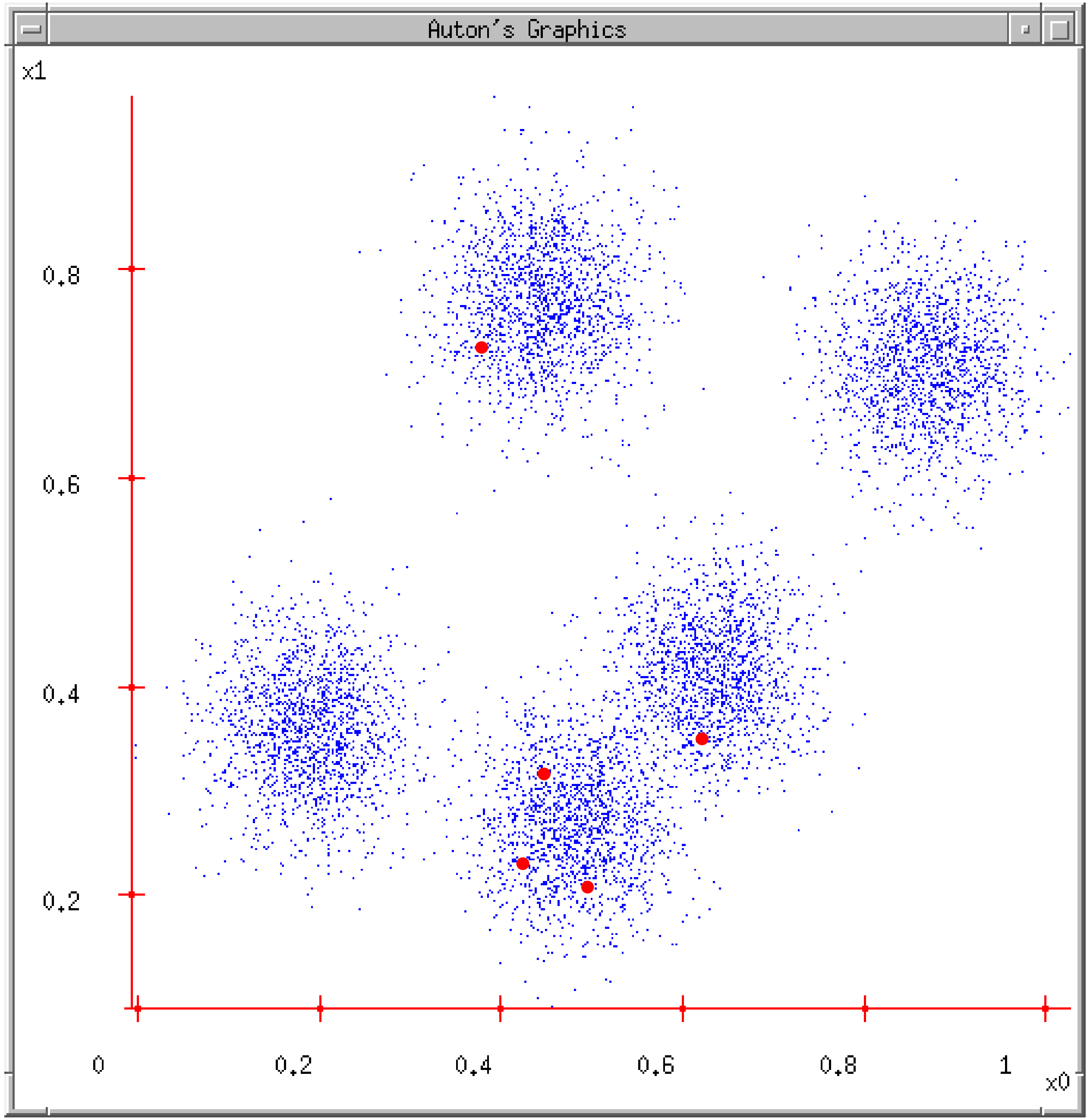
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



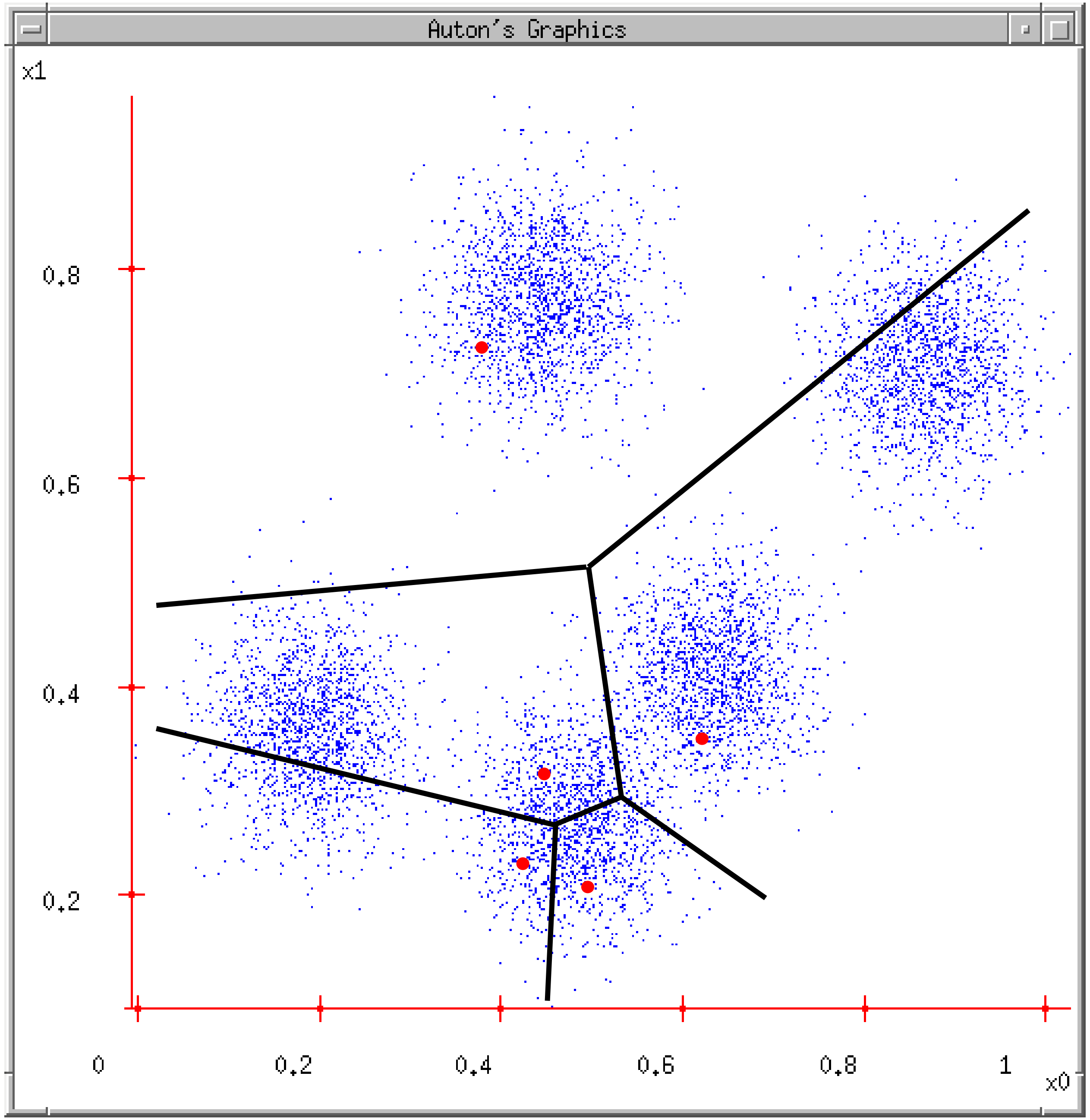
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



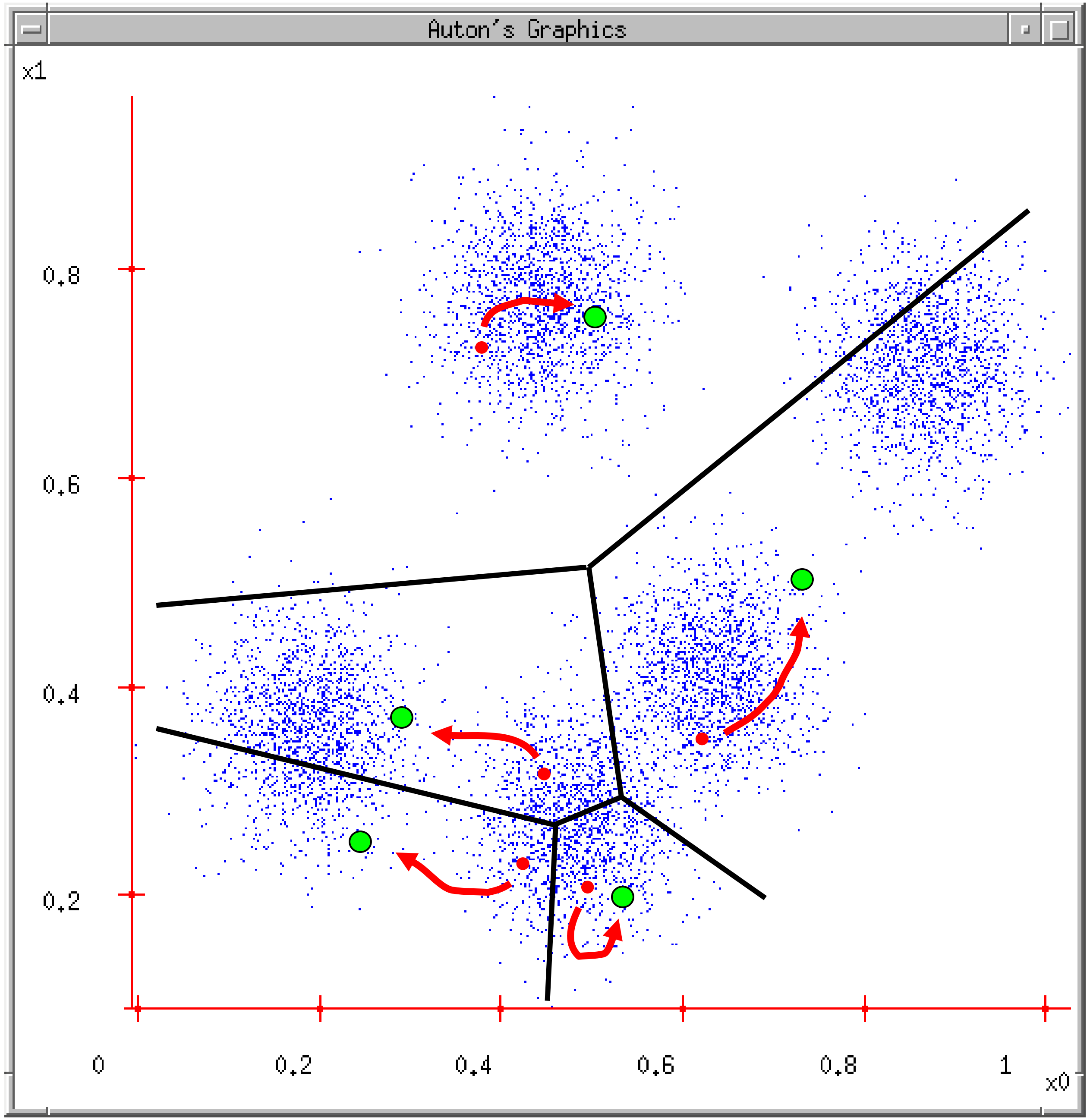
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



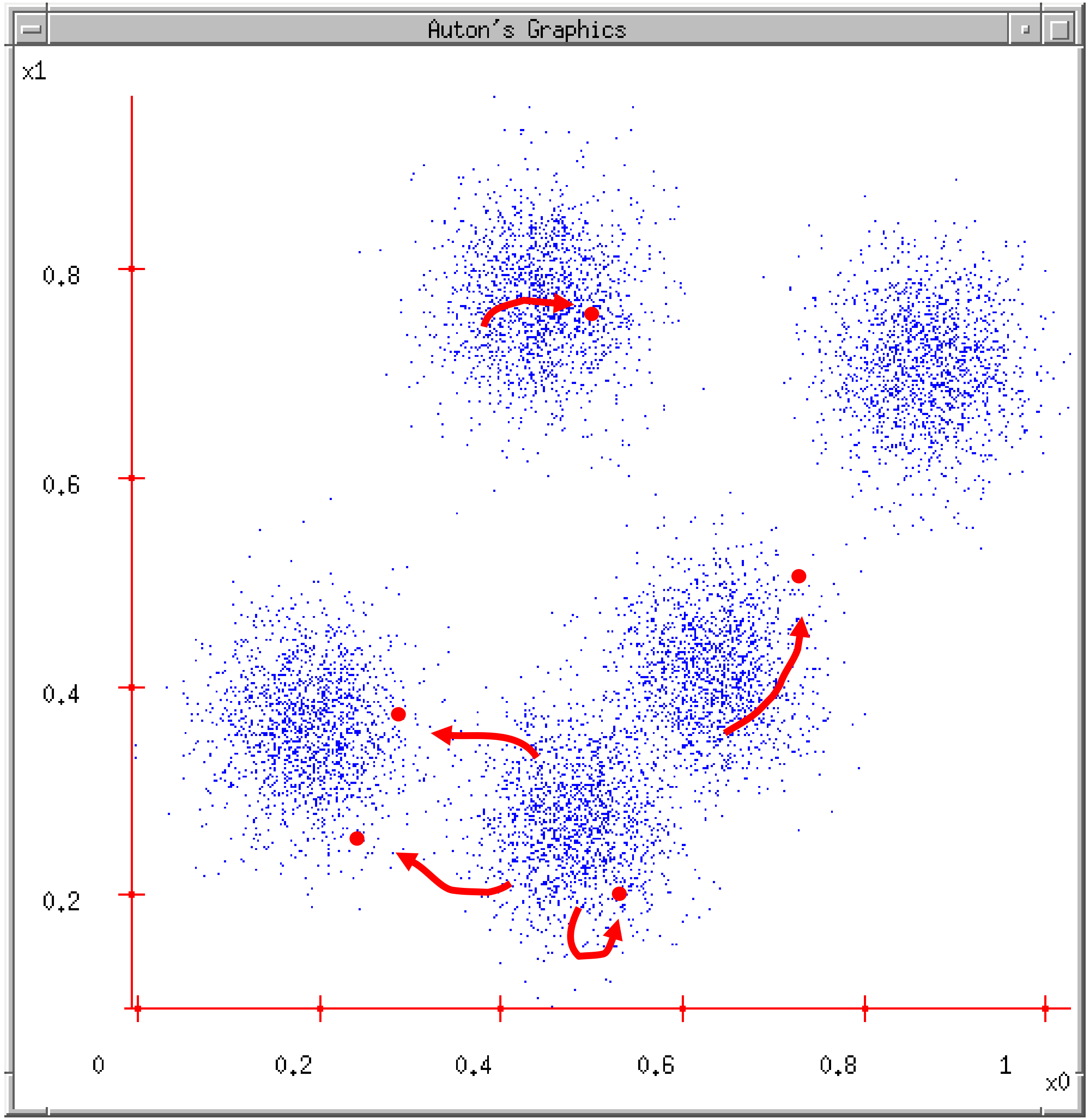
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns

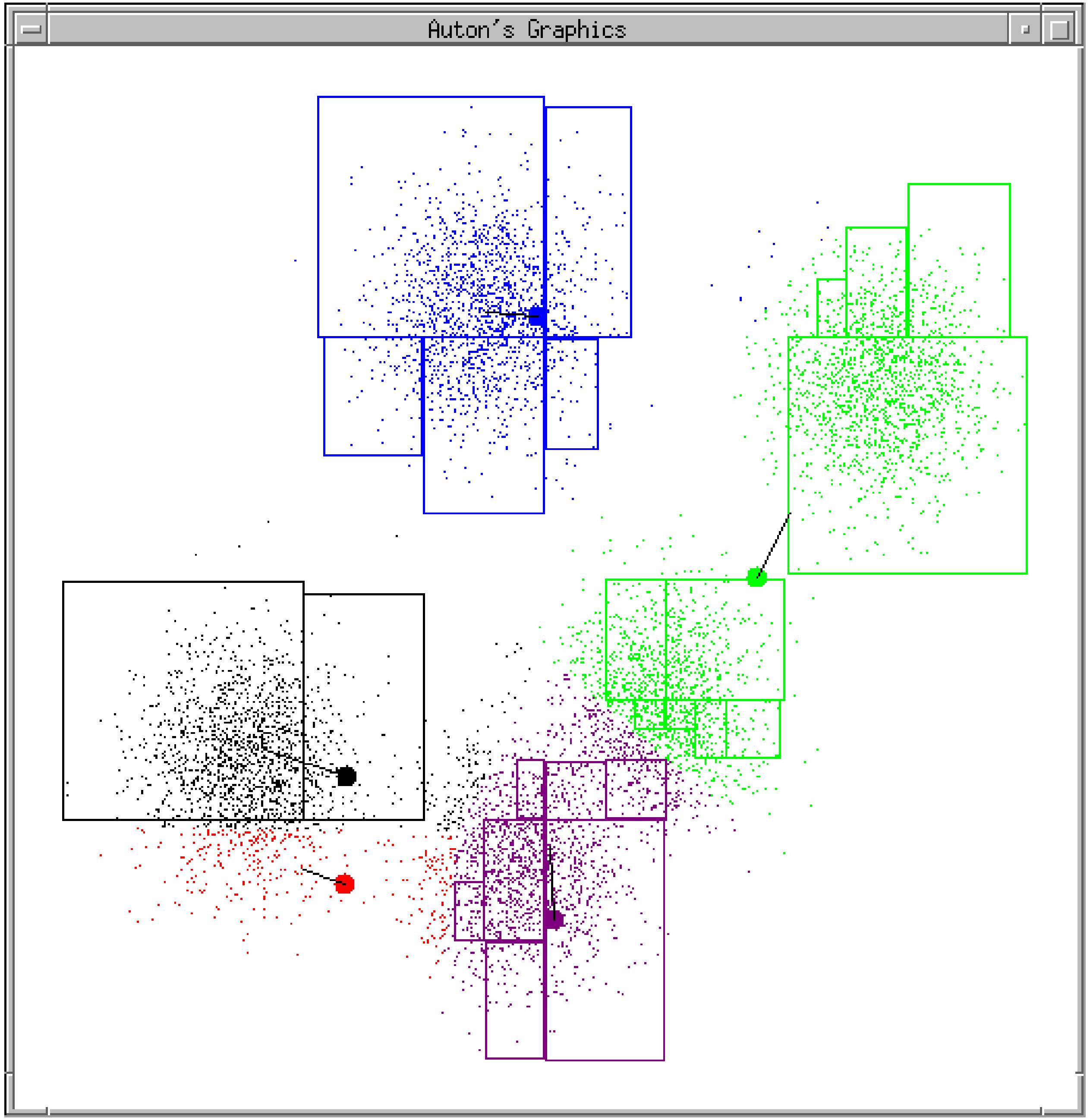


K-means

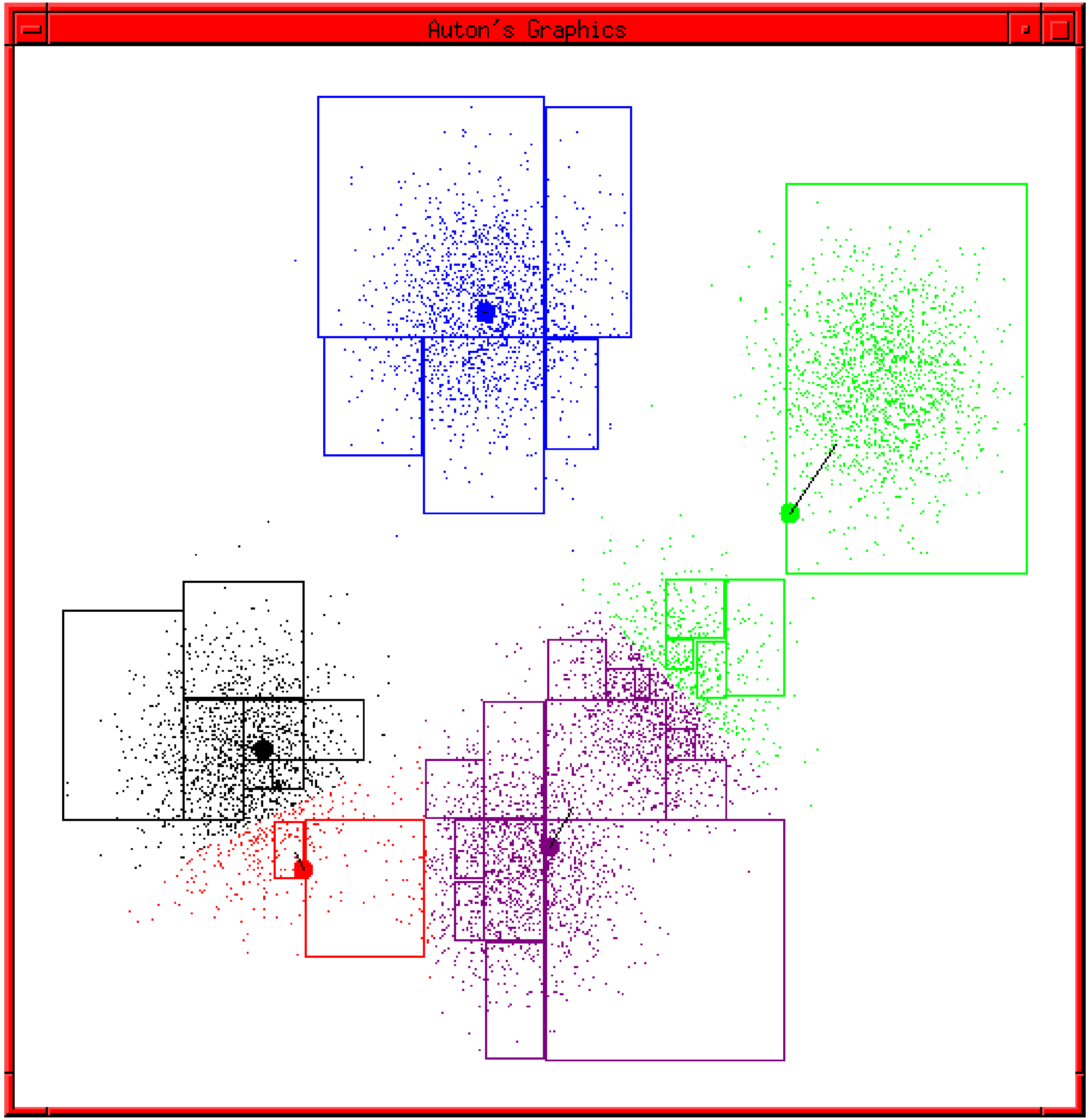
1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



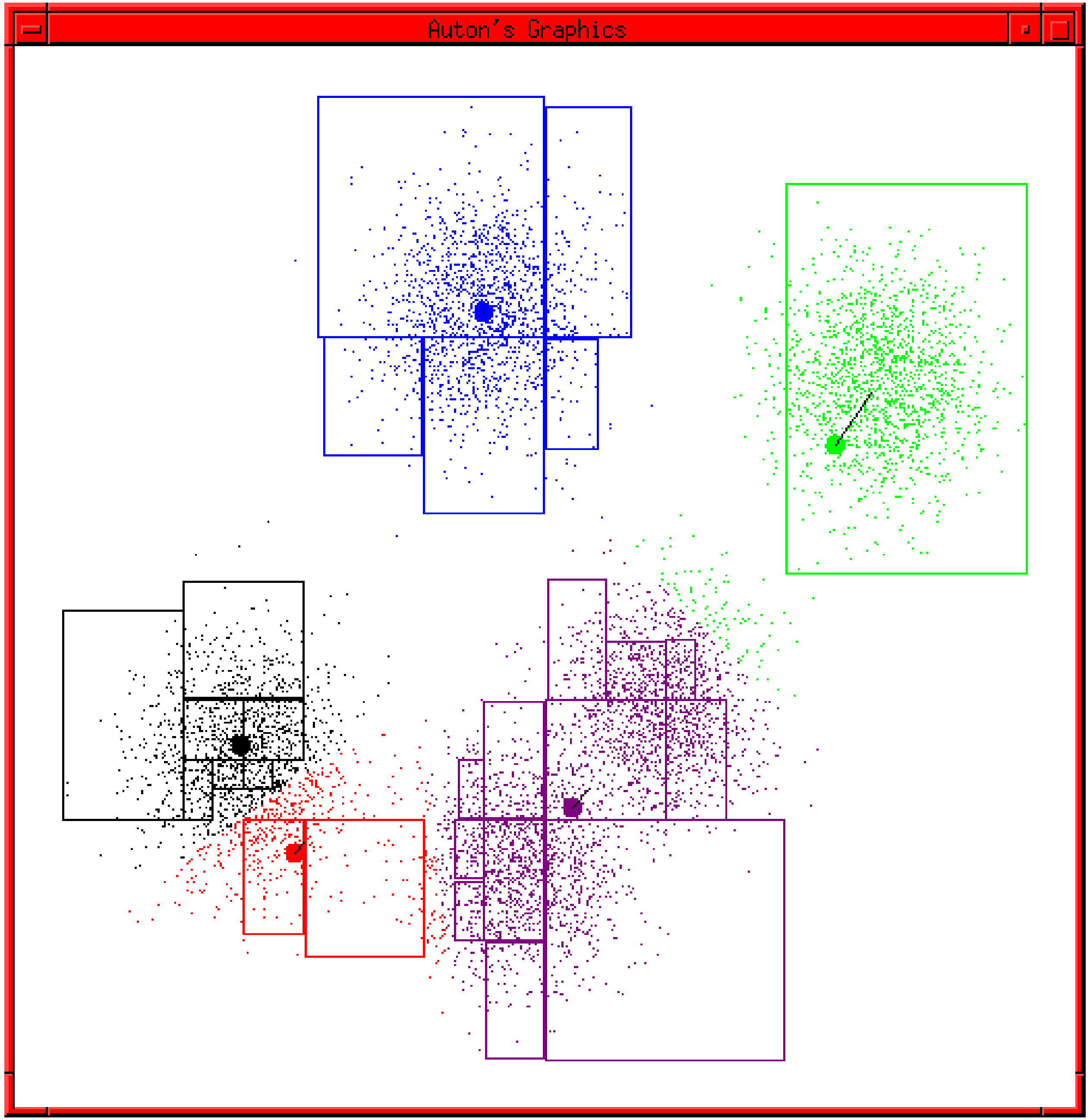
K-means continues …



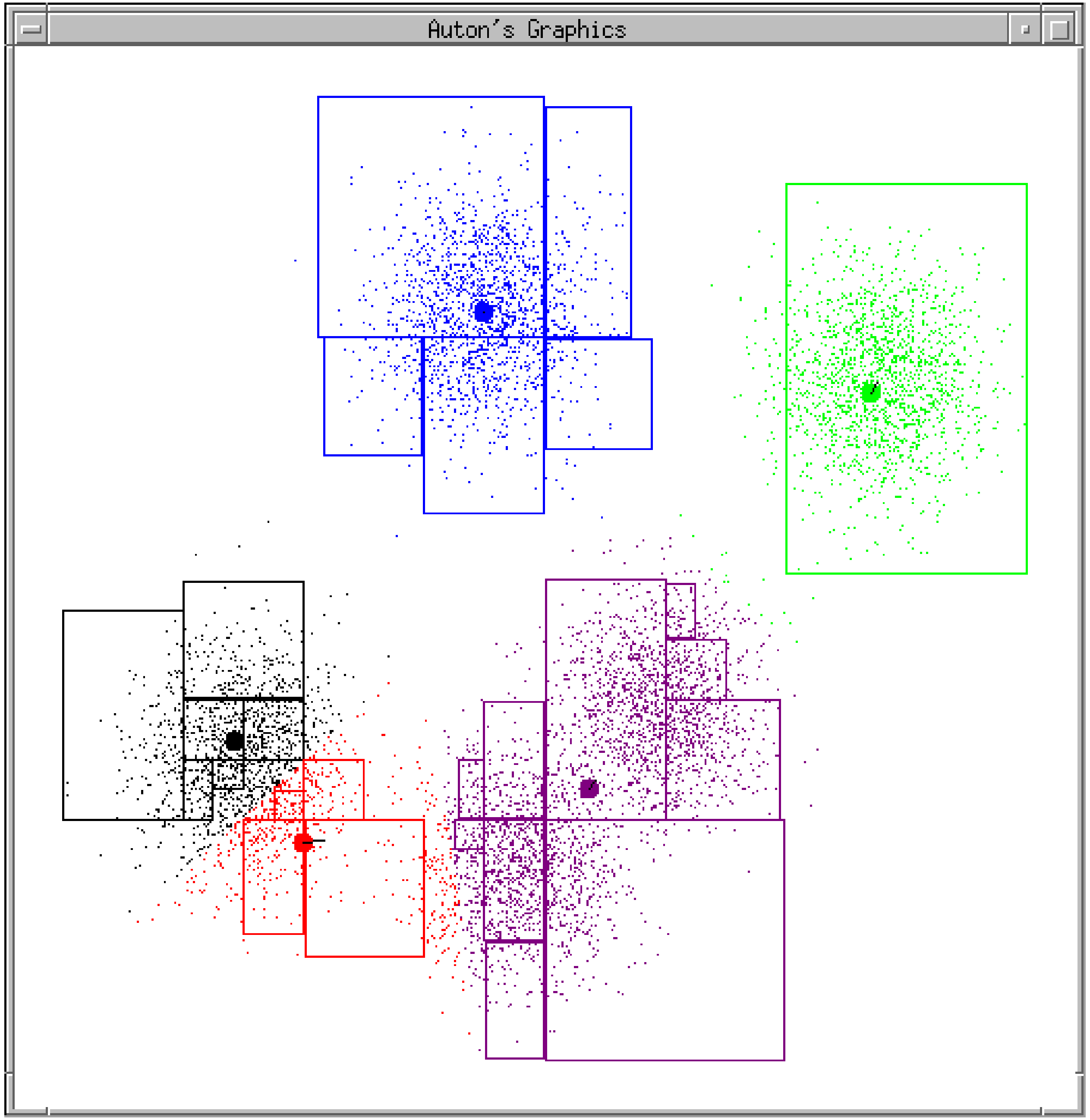
K-means continues …



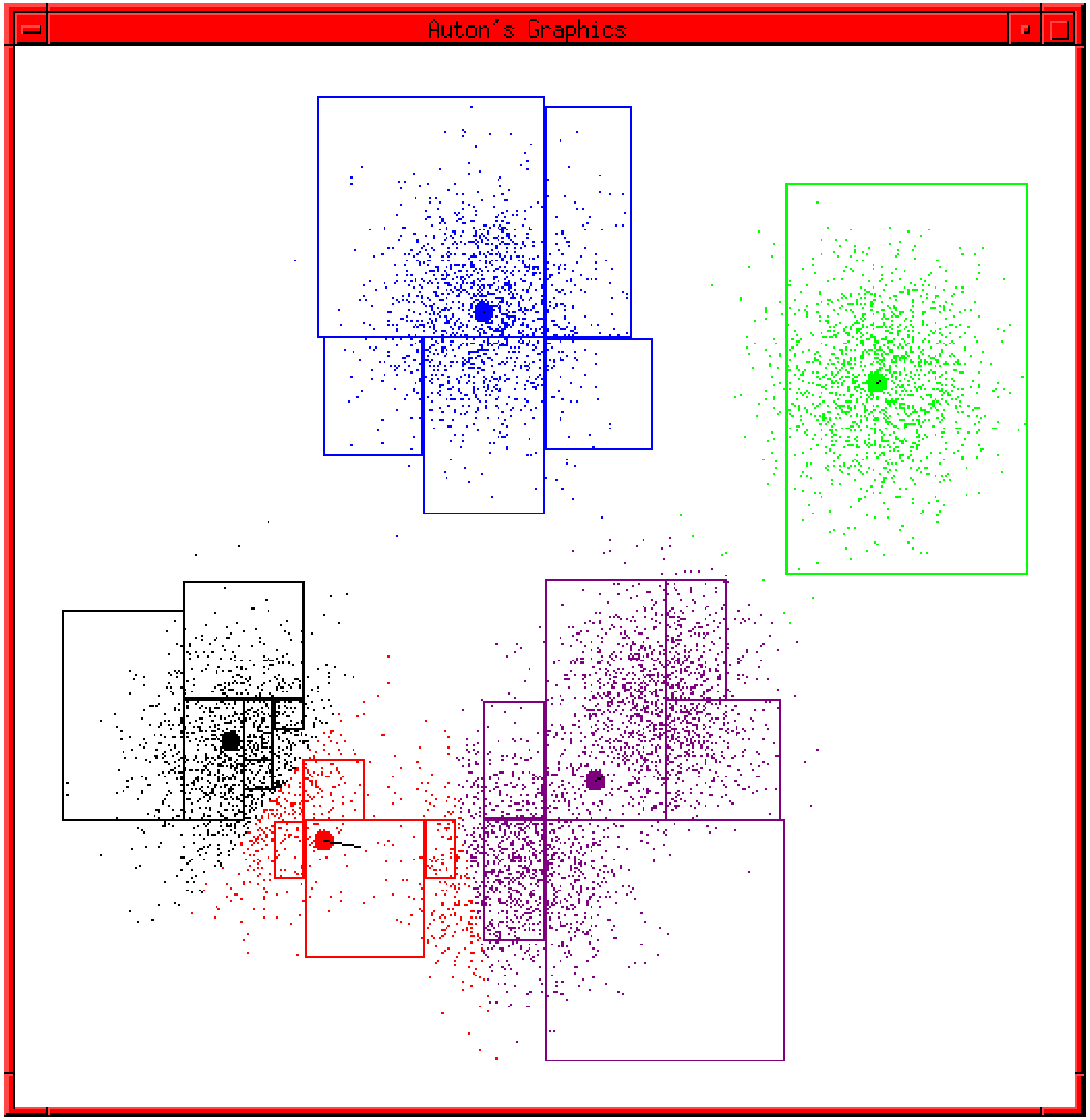
K-means continues …



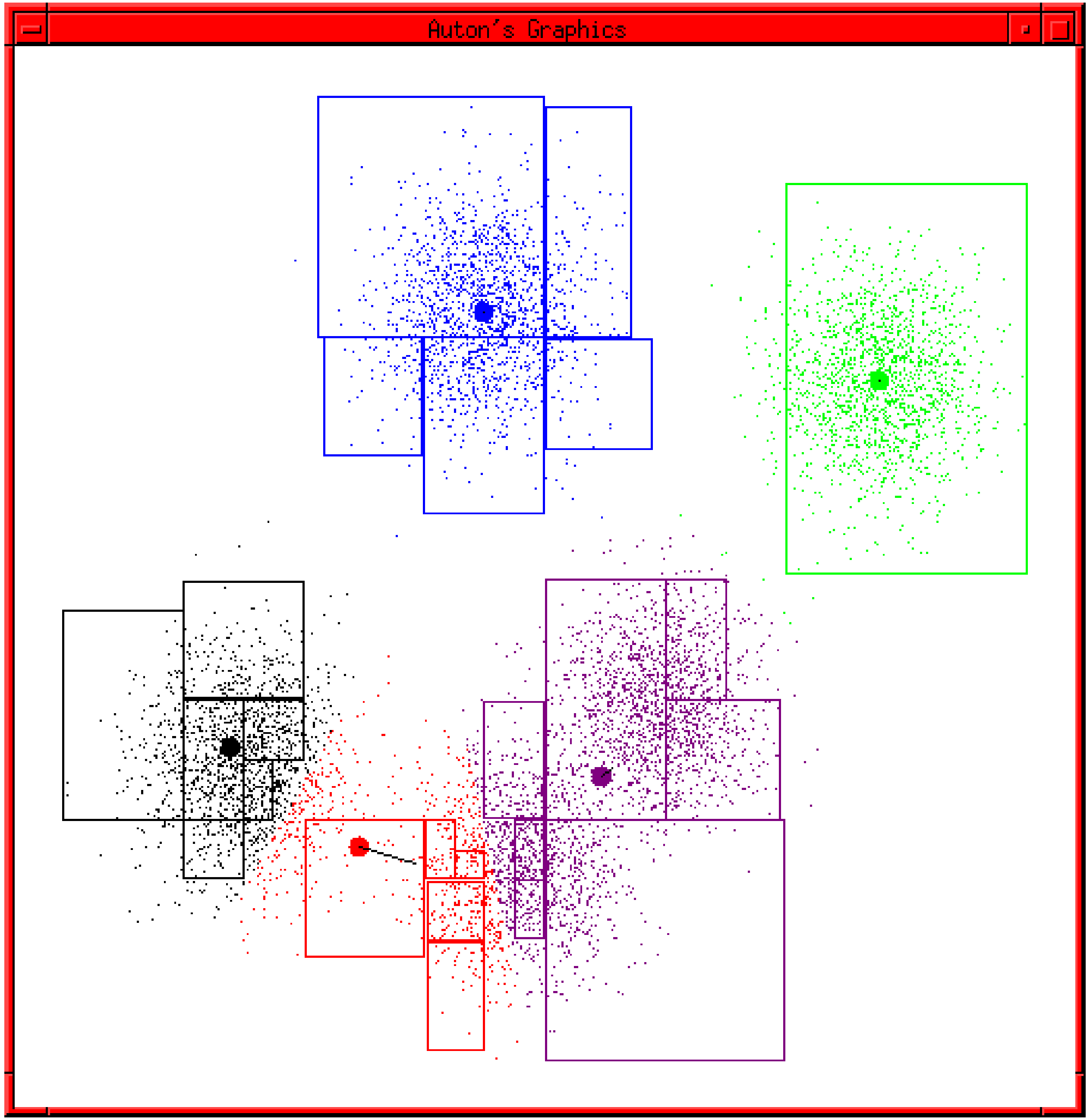
K-means continues …



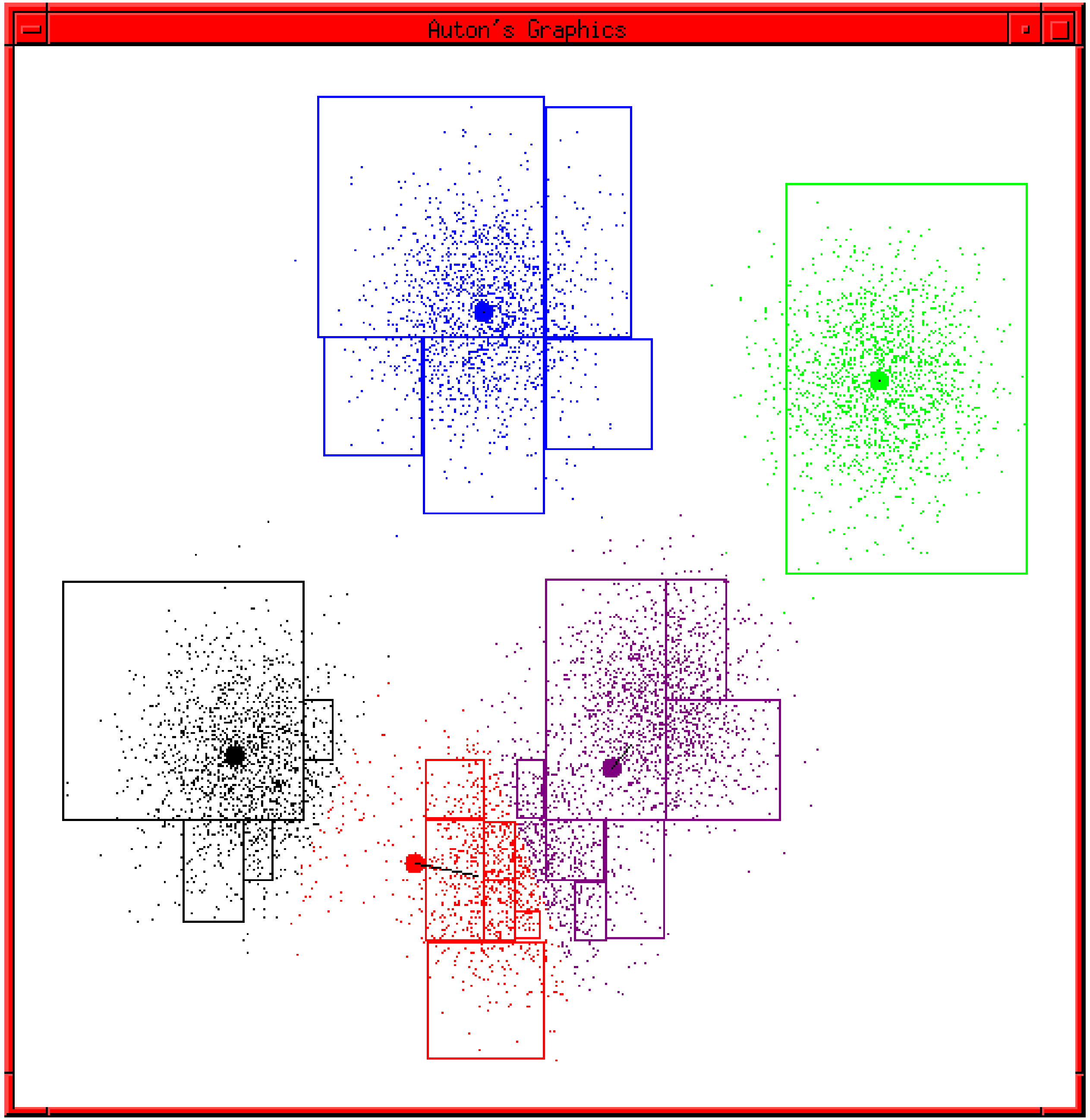
K-means continues …



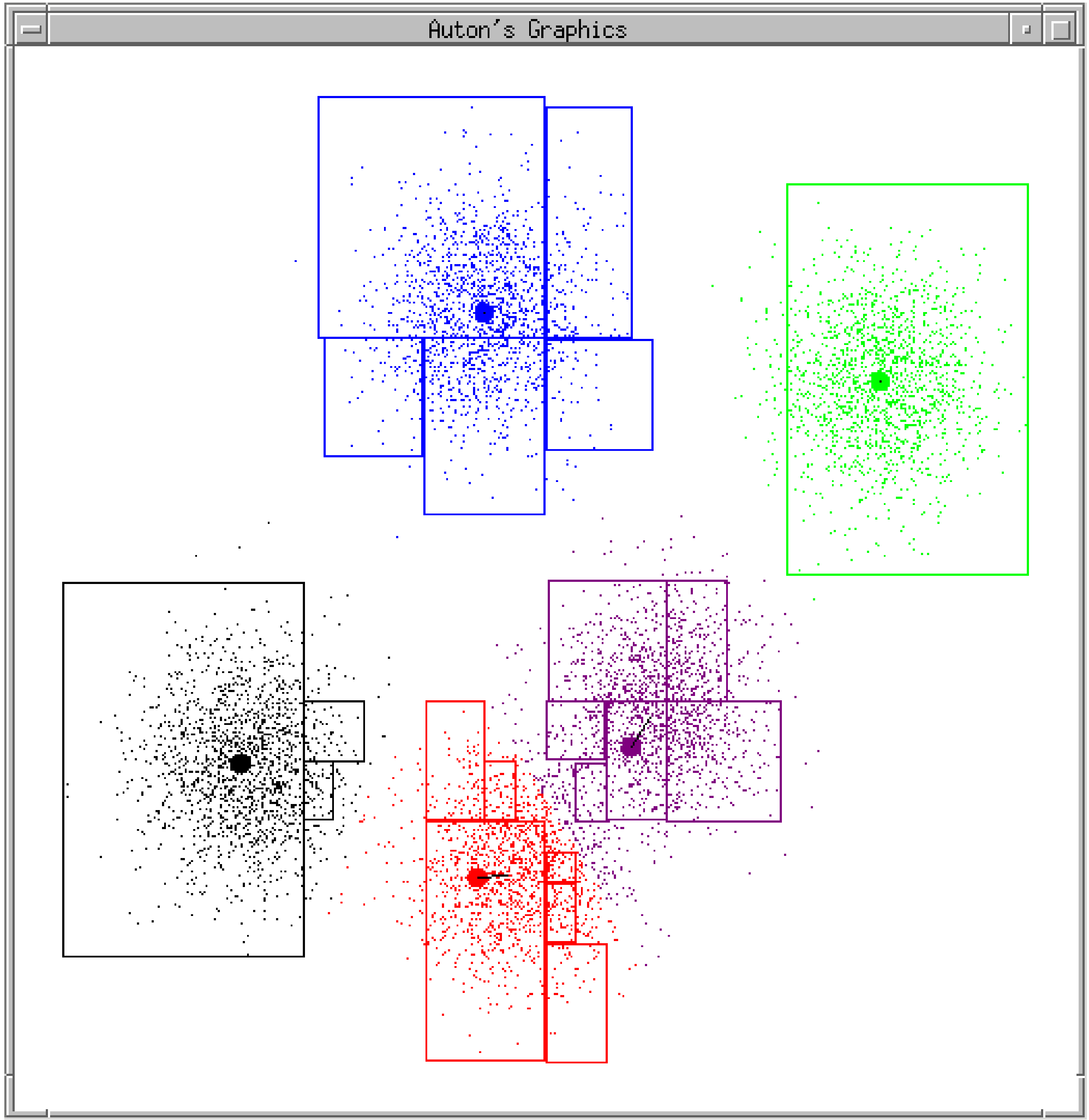
K-means continues …



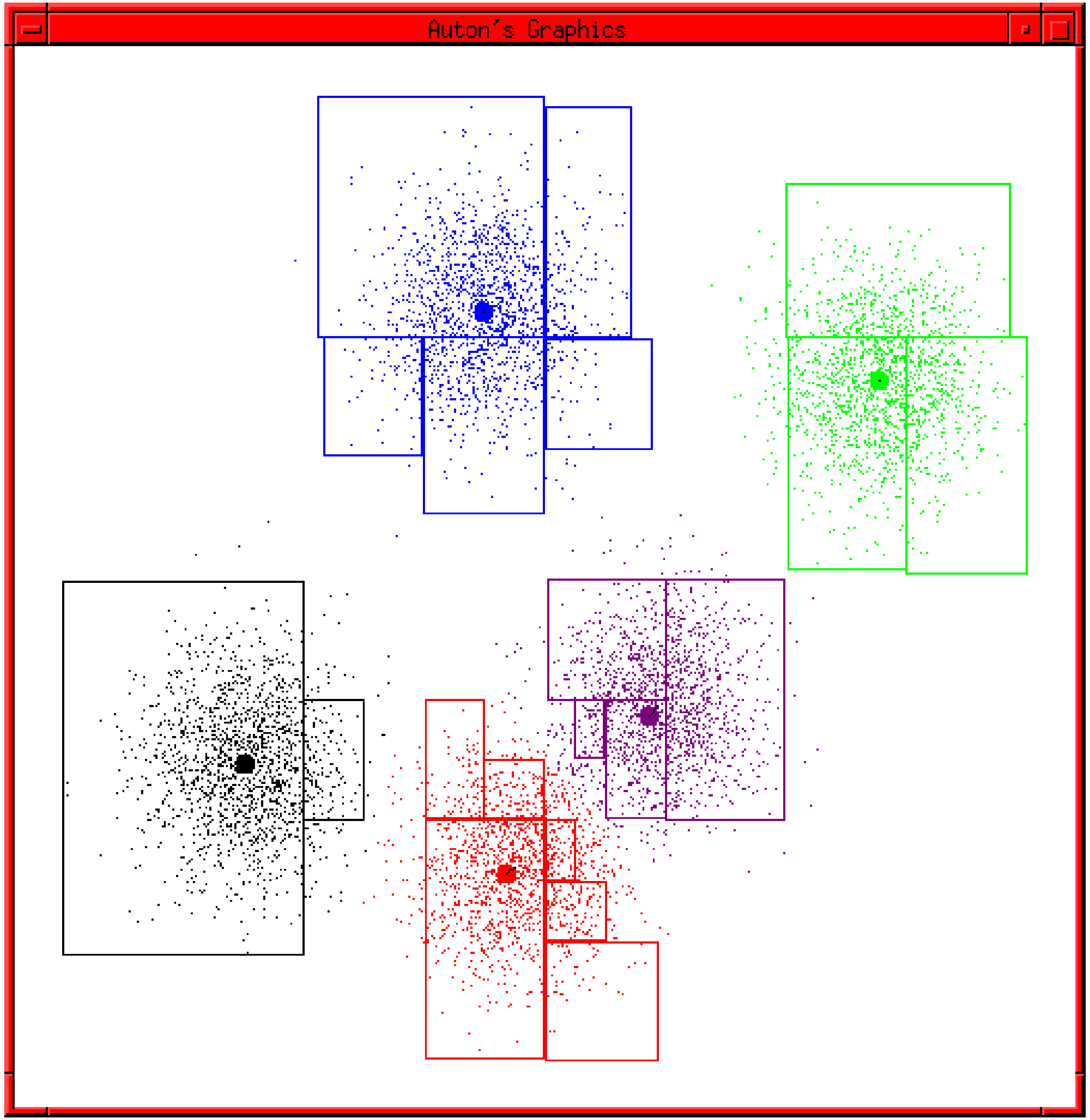
K-means continues …



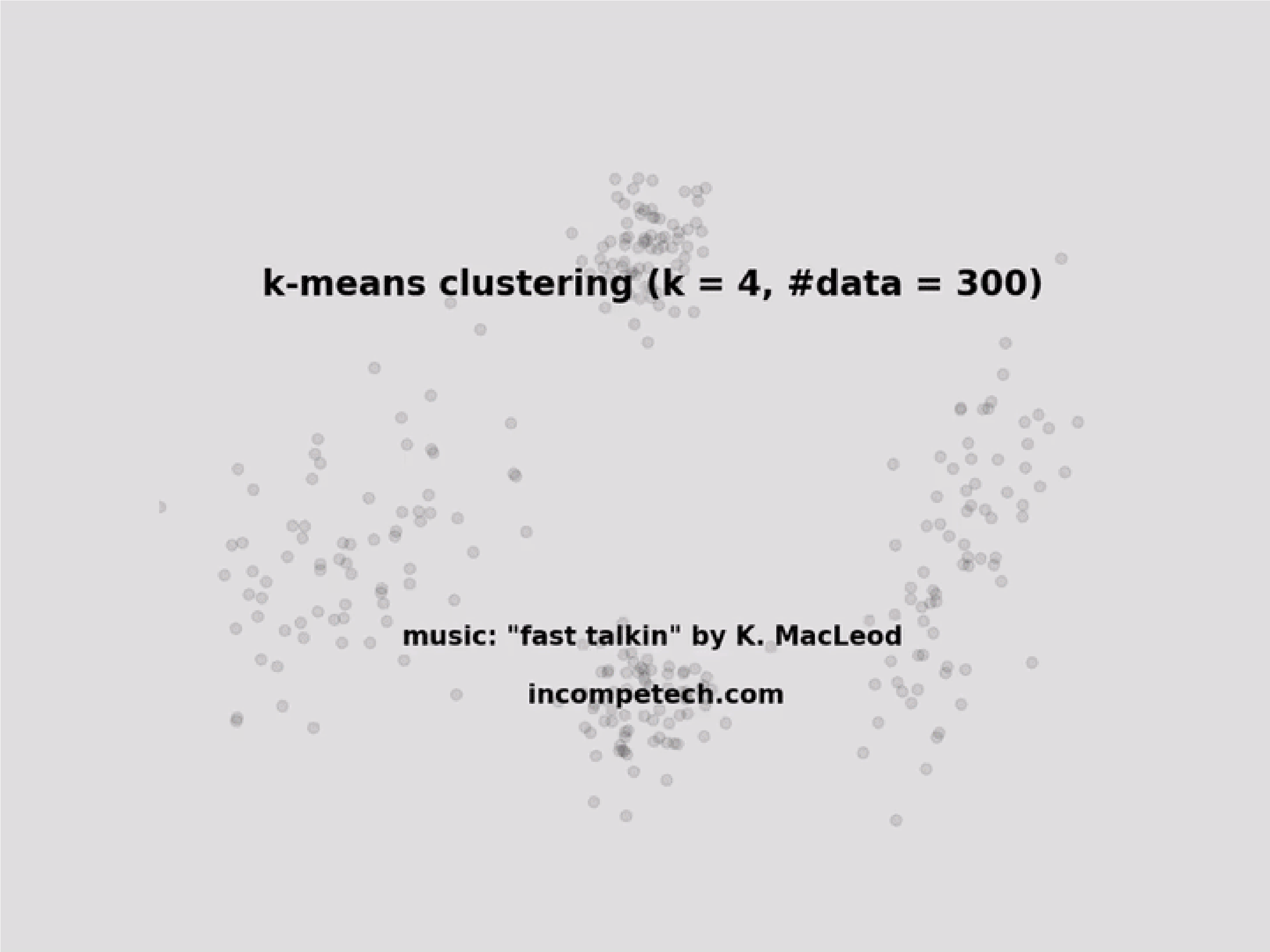
K-means continues …



K-means terminates



Animation


k-means clustering (k = 4, #data = 300)
music: "fast talkin" by K. MacLeod
incompetech.com

Step 1: Data assignment step

- Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if c_i is the collection of centroids in set C, then each data point x is assigned to a cluster based on

$$\underset{c_i \in C}{\arg\min}\ dist(c_i, x)^2$$

- where $dist(\cdot)$ is the standard (L_2) Euclidean distance. Let the set of data point assignments for each i^{th} cluster centroid be S_i.

Step 2: Centroid update step

- In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

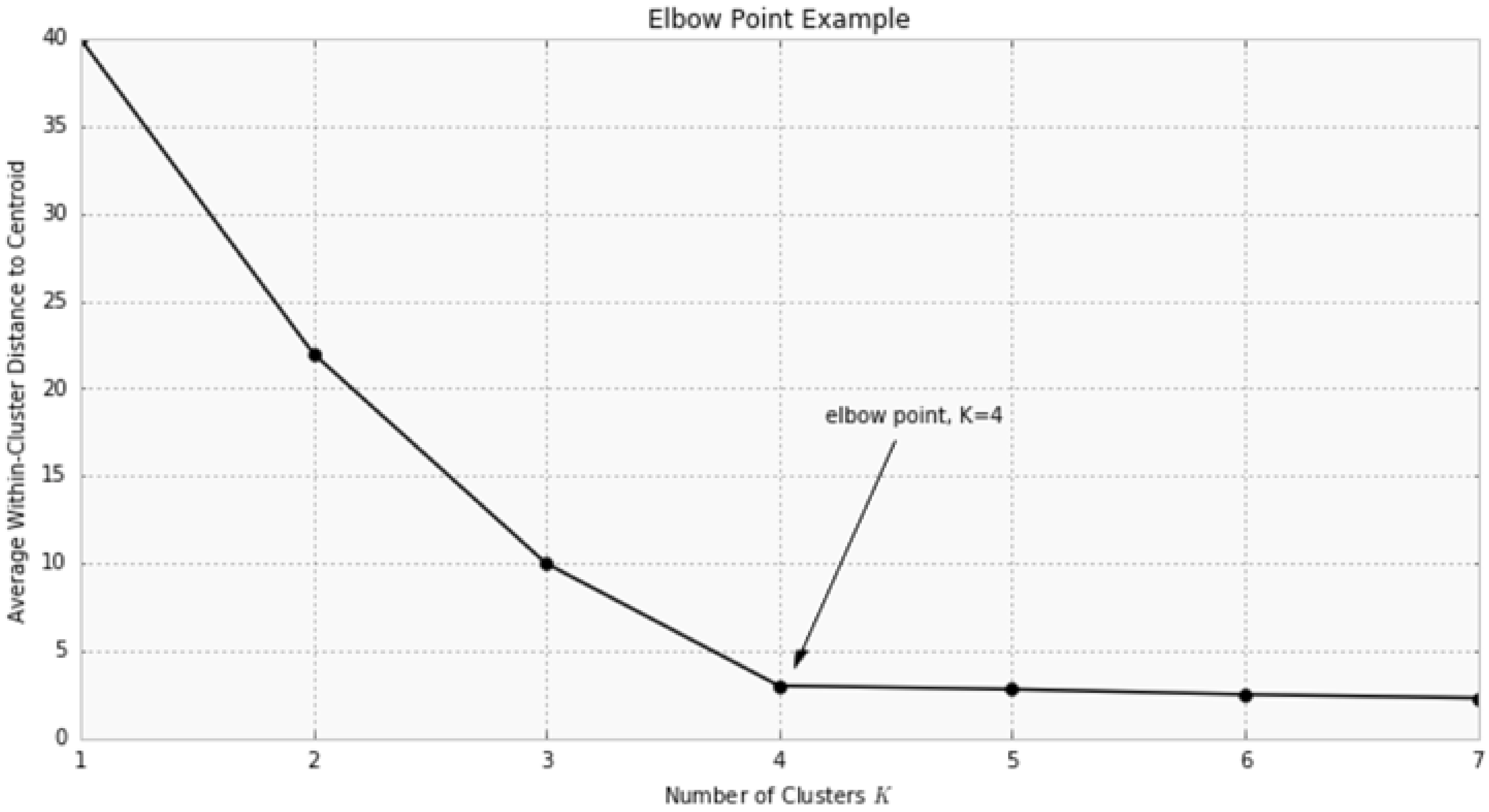
$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$

- The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached).
- This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome.

Choosing K

- One of the metrics that is commonly used to compare results across different values of *K* is the mean distance between data points and their cluster centroid.
- Since increasing the number of clusters will always reduce the distance to data points, increasing *K* will *always* decrease this metric, to the extreme of reaching zero when *K* is the same as the number of data points.
- Thus, this metric cannot be used as the sole target. Instead, mean distance to the centroid as a function of *K* is plotted and the "elbow point," where the rate of decrease sharply shifts, can be used to roughly determine *K*.

Choosing K



Elbow Point Example
Average Within-Cluster Distance to Centroid
Number of Clusters K
elbow point, K=4
0
5
10
15
20
25
30
35
40
1
2
3
4
5
6
7

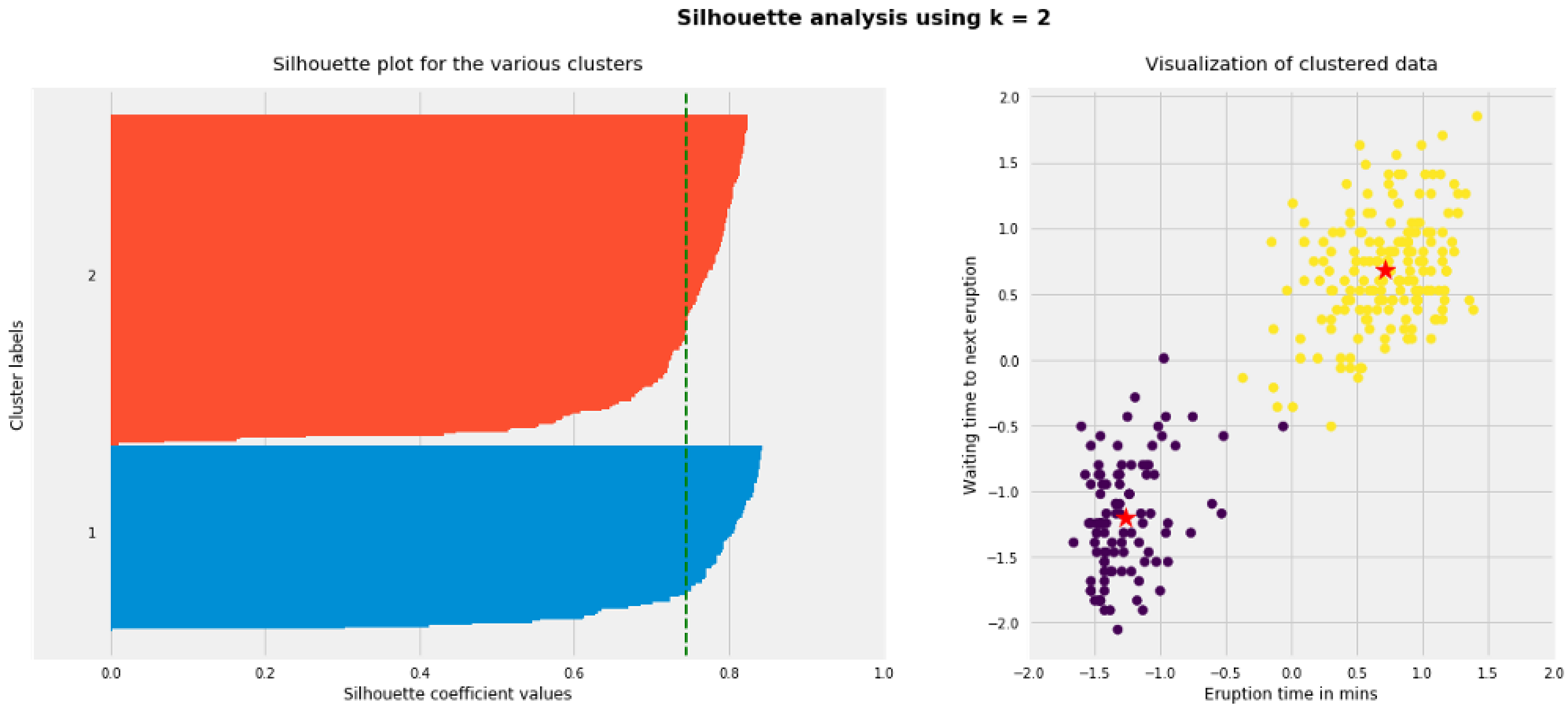
Clustering Evaluation

- **Silhouette analysis** can be used to determine the degree of separation between clusters. For each sample:
- Compute the average distance from all data points in the same cluster (ai).
- Compute the average distance from all data points in the closest cluster (bi).
- Compute the coefficient:

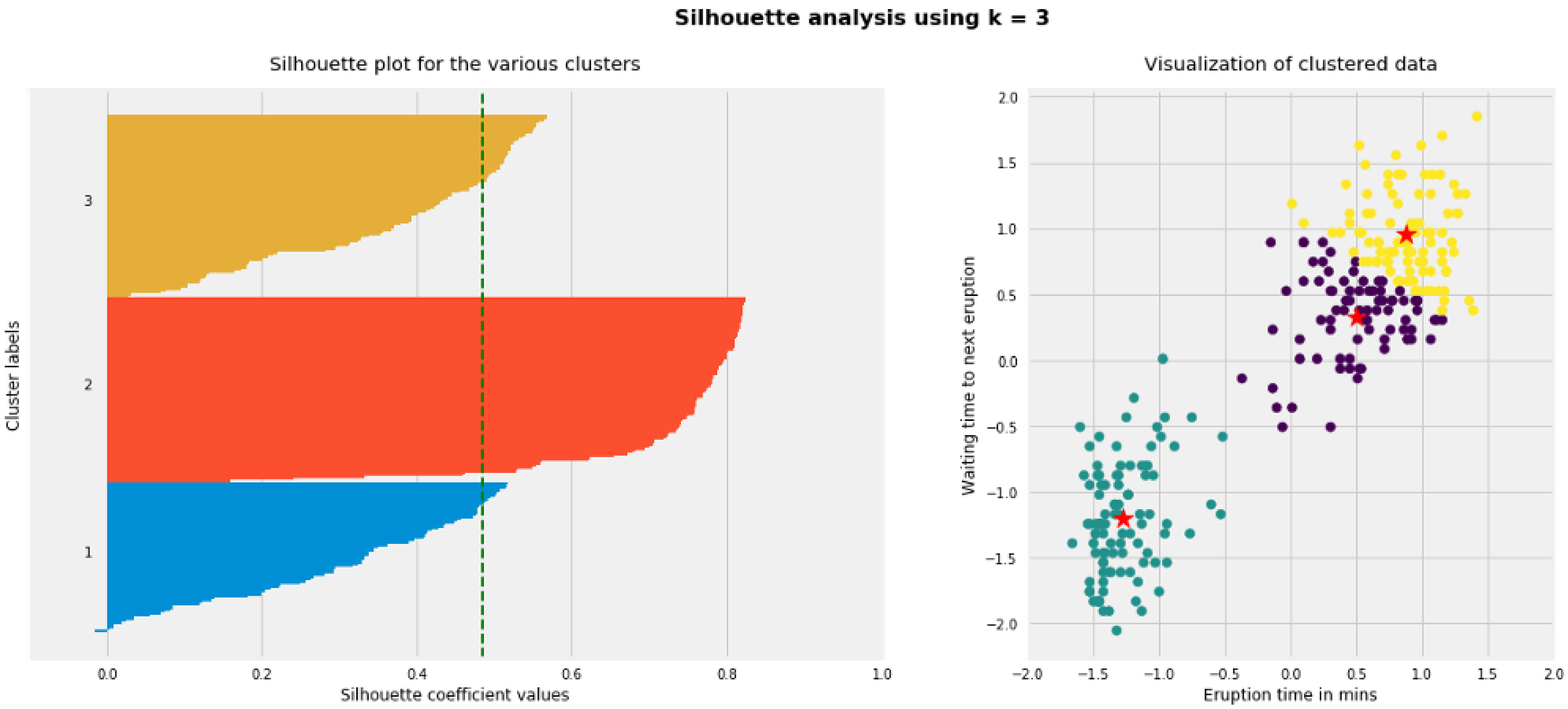
$$\frac{b^i - a^i}{max(a^i, b^i)}$$

- The coefficient can take values in the interval [-1, 1].
  - If it is 0 –> the sample is very close to the neighboring clusters.
  - It it is 1 –> the sample is far away from the neighboring clusters.
  - It it is -1 –> the sample is assigned to the wrong clusters.
- Therefore, we want the coefficients to be as big as possible and close to 1 to have a good clusters. We'll use here geyser dataset again because its cheaper to run the silhouette analysis and it is actually obvious that there is most likely only two groups of data points.

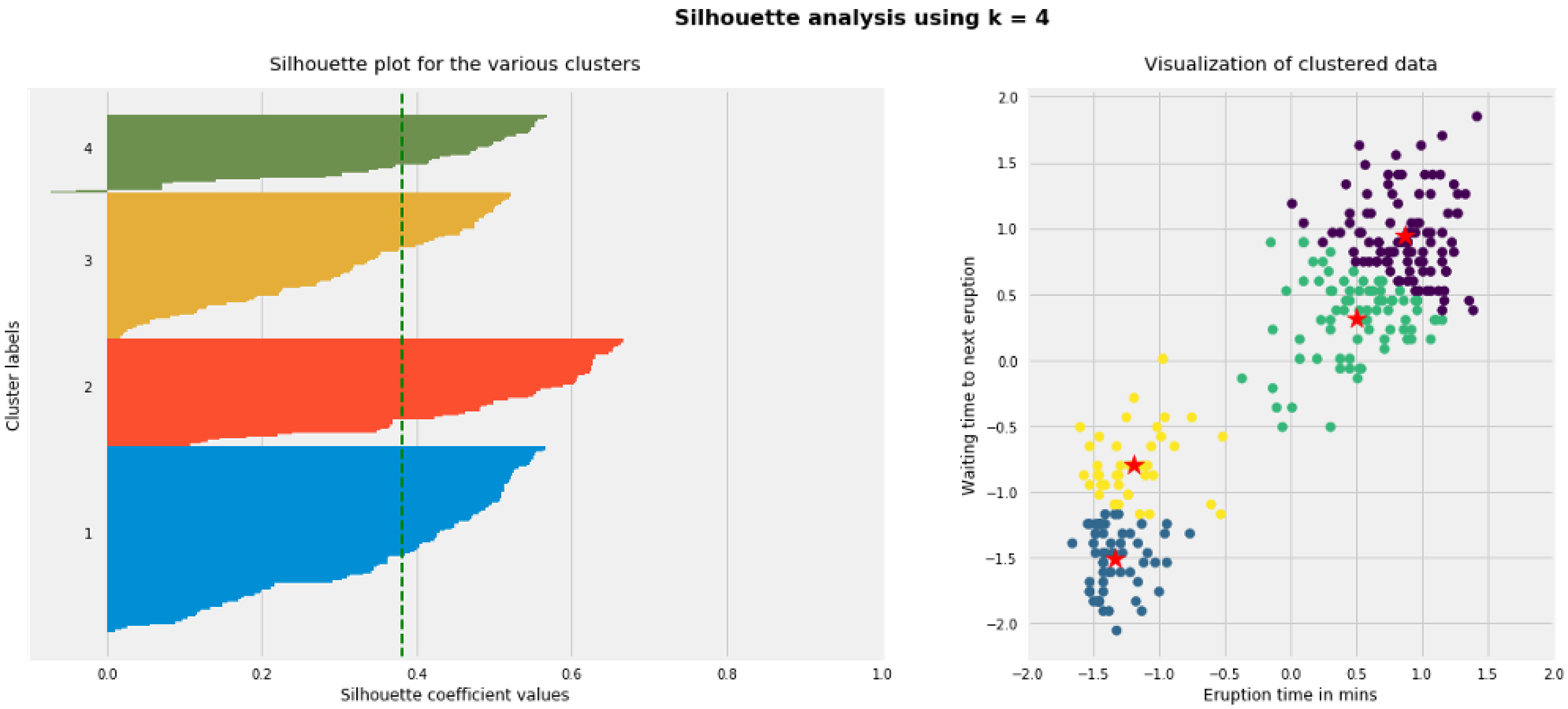
Silhouette analysis



Silhouette analysis



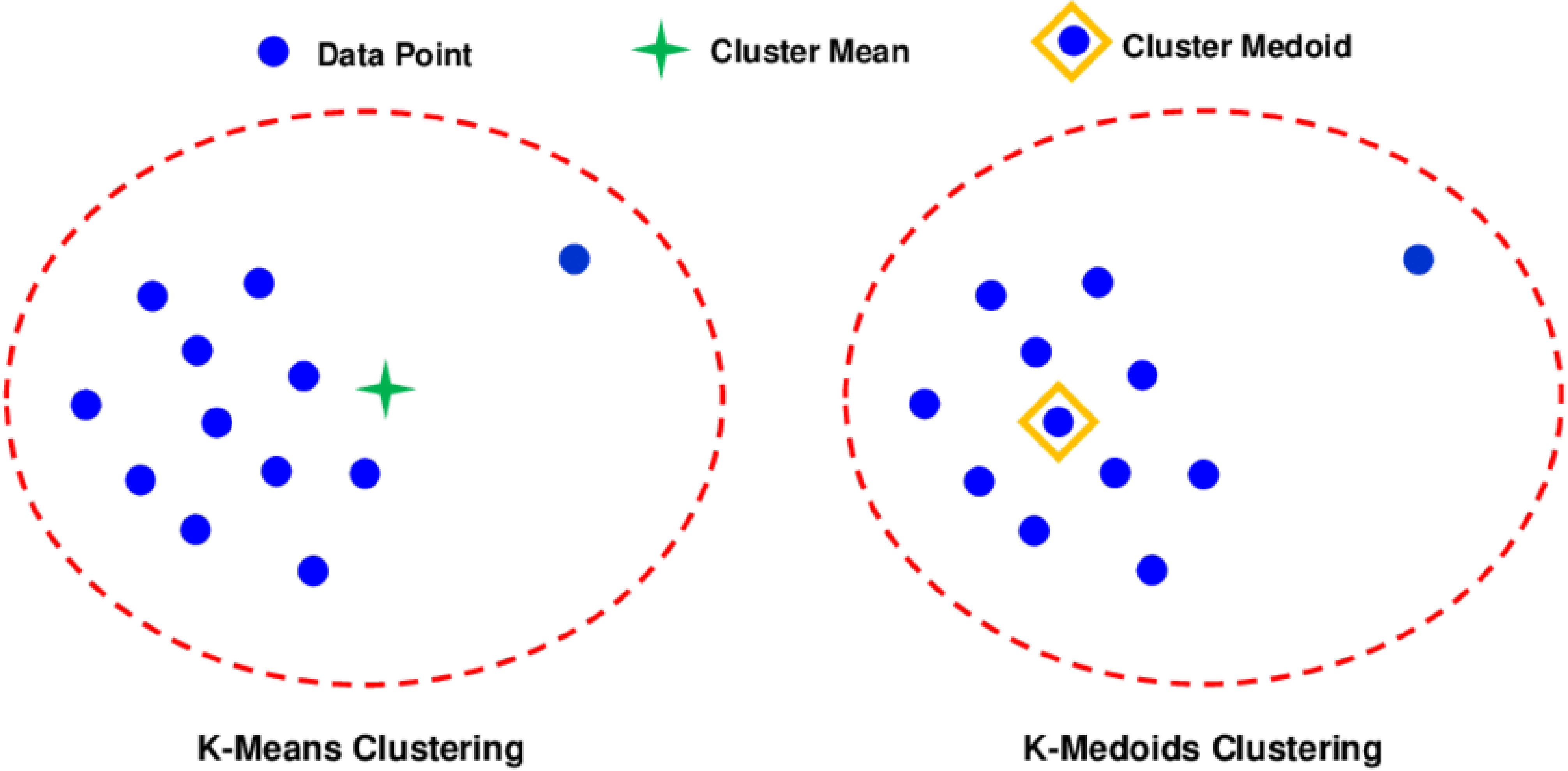
Silhouette analysis



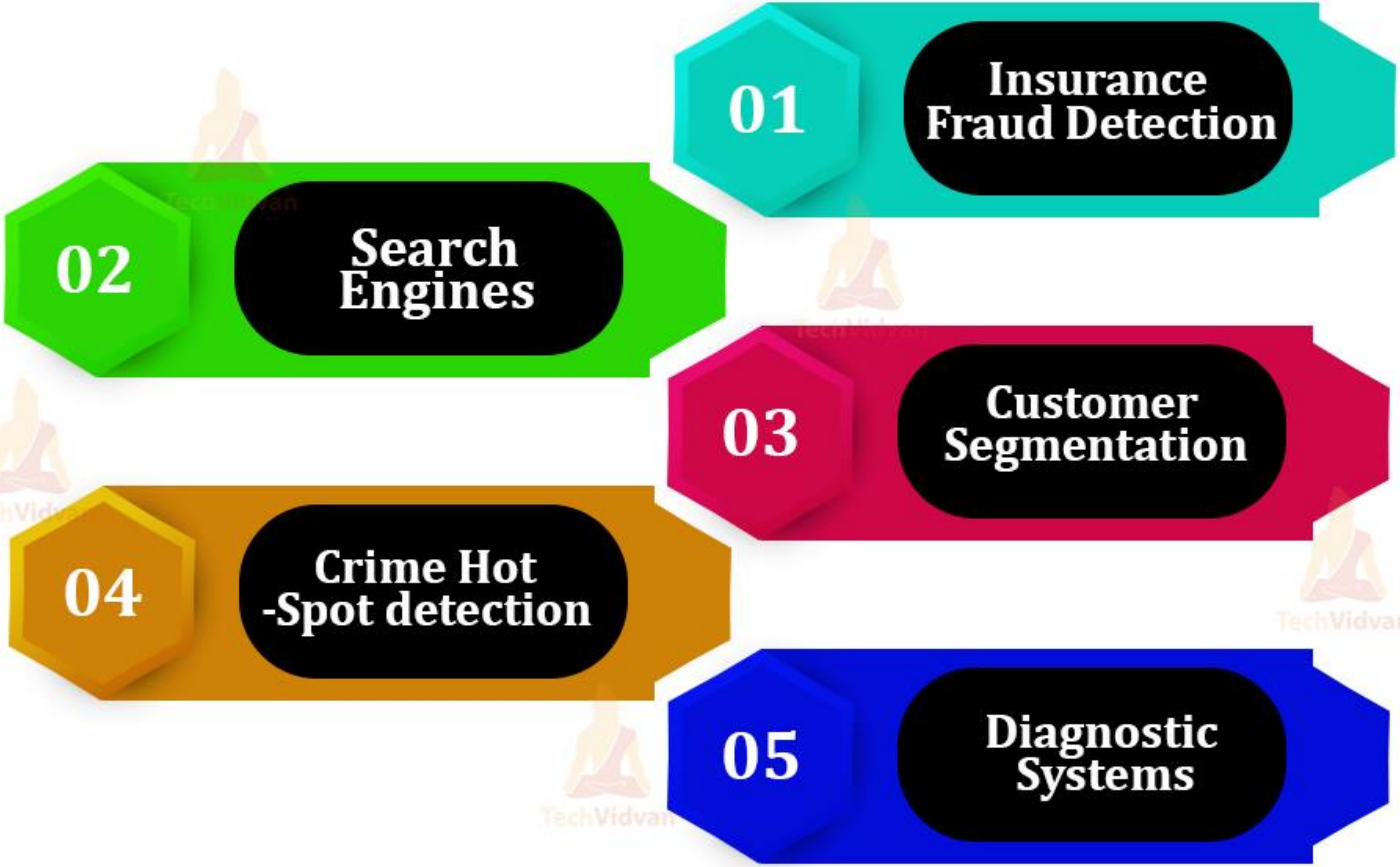
Silhouette analysis

- As the above plots show, n_clusters=2 has the best average silhouette score of around 0.75 and all clusters being above the average shows that it is actually a good choice. Also, the thickness of the silhouette plot gives an indication of how big each cluster is. The plot shows that cluster 1 has almost double the samples than cluster 2. However, as we increased n_clusters to 3 and 4, the average silhouette score decreased dramatically to around 0.48 and 0.39 respectively. Moreover, the thickness of silhouette plot started showing wide fluctuations. The bottom line is: Good n_clusters will have a well above 0.5 silhouette average score as well as all of the clusters have higher than the average score.

K-Medoids Algorithm



Application of K-means Clustering in ML



Happy Pongal 2023