K-Means Algorithm for Clustering

Liang Liang

Categories of Machine Learning

- Supervised Learning

to model the relationship between measured features of data and some label associated with the data

- Unsupervised Learning

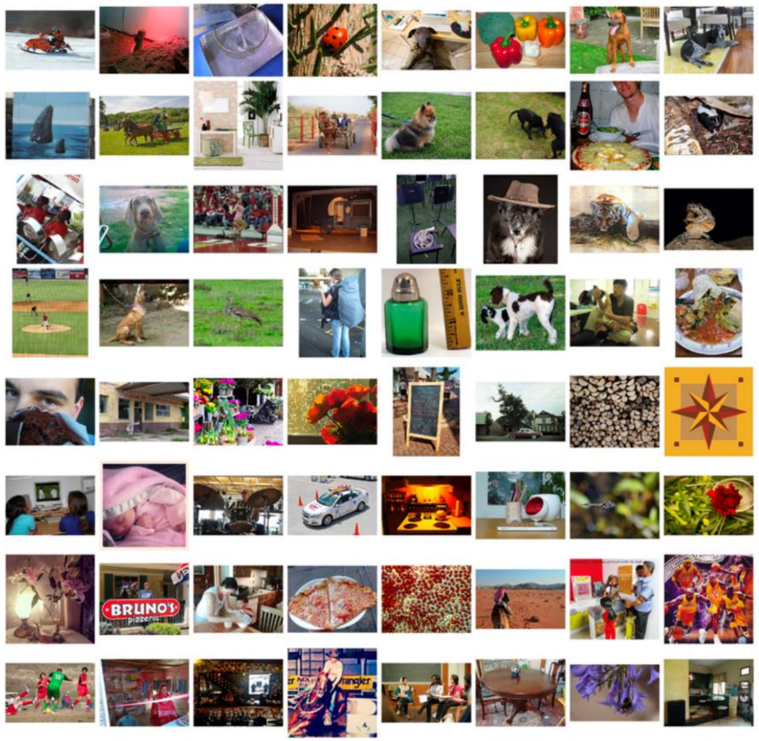
to model the features of a dataset without reference to any label

- Reinforcement Learning

the goal is to develop a model (agent) that improves its performance based on interactions with the environment

Cluster images

Image
Dataset

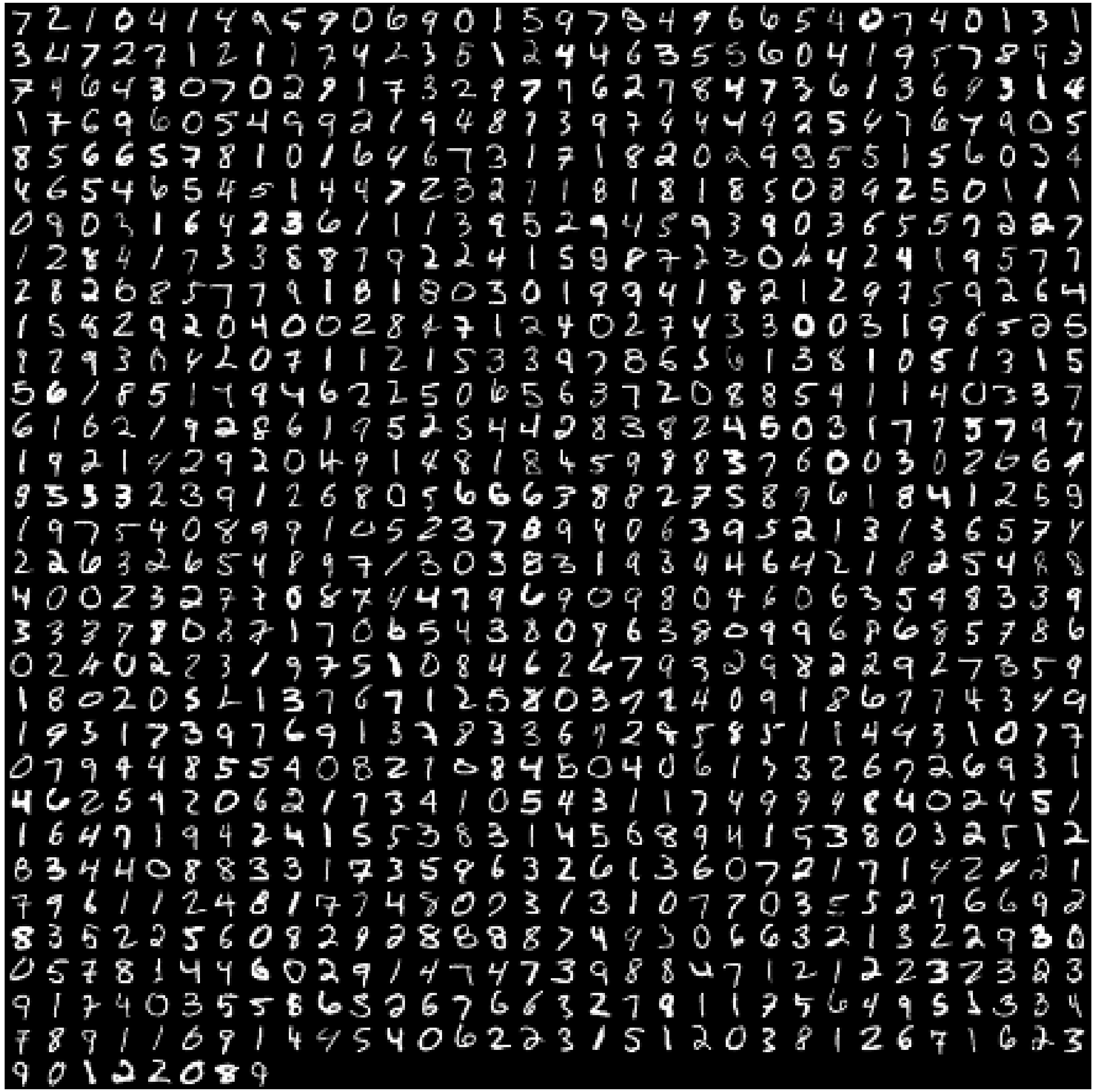


clustering

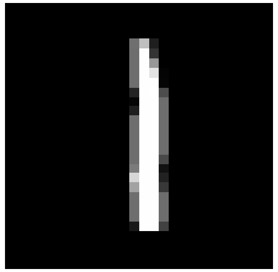
Goal of clustering:
Divide objects into groups, and objects within a group are more similar than those outside the group

unsupervised learning

Clustering handwritten digit images



image



Clustering the images into ten groups/clusters

A cluster may correspond to a digit.

Cluster customers - customer segmentation

- Assume you work in the credit card department of a bank
  you job title is data scientist

- to understand the behaviors of the customers (credit card holders) and improve marketing strategies, you may need to categorize the customers based on their characteristics (income, age, buying behavior, etc).

- Find the clusters/groups that contain valuable customers:
  e.g., high income but low annual spend.

2.3. Clustering

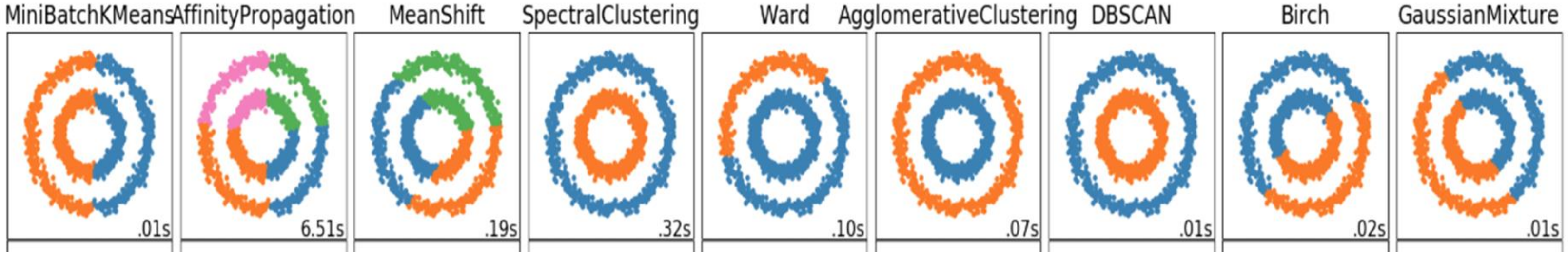
Clustering of unlabeled data can be performed with the module `sklearn.cluster`.

Each clustering algorithm comes in two variants: a class, that implements the `fit` method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the `labels_` attribute.

Input data

One important thing to note is that the algorithms implemented in this module can take different kinds of matrix as input. All the methods accept standard data matrices of shape `[n_samples, n_features]`. These can be obtained from the classes in the `sklearn.feature_extraction` module. For `AffinityPropagation`, `SpectralClustering` and `DBSCAN` one can also input similarity matrices of shape `[n_samples, n_samples]`. These can be obtained from the functions in the `sklearn.metrics.pairwise` module.

2.3.1. Overview of clustering methods



sklearn.cluster.KMeans

class sklearn.cluster.**KMeans**(*n_clusters=8, init='k-means++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='auto', verbose=0, random_state=None, copy_x=True, n_jobs=None, algorithm='auto'*) [source]

K-Means clustering.

Read more in the User Guide.

Parameters:

n_clusters : *int, default=8*

The number of clusters to form as well as the number of centroids to generate.

init : *{'k-means++', 'random'} or ndarray of shape (n_clusters, n_features), default='k-means++'*

Method for initialization, defaults to 'k-means++':

'k-means++' : selects initial cluster centers for k-mean clustering in a smart way to speed up convergence. See section Notes in k_init for more details.

'random': choose k observations (rows) at random from data for the initial centroids

K-means Algorithm for Clustering Objects

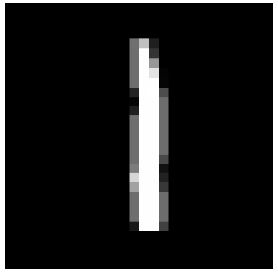
- Represent each object by a numerical vector

- Input to the k-means algorithm is a set of vectors
  we need to put those vectors into a 2D array (matrix/table)

- Output from the k-means algorithm is a set of clusters (groups)
  each cluster contains a subset of the vectors/objects
  the clusters are disjoint (do not share any vectors/objects)

- Clustering is based on the distance between two vectors
  we need a function to measure the distance(vectorA, vectorB)

Represent an image by a vector



This image has 28×28 pixels.

It is a matrix/ 2D array $A \in \mathbb{R}^{28\times 28}$

$$A=\begin{array}{cc} \begin{bmatrix} A_{0,0} & \cdots & A_{0,27} \\ \cdots & \cdots & \cdots \\ A_{27,0} & \cdots & A_{27,27} \end{bmatrix} & \begin{array}{l} \boxed{\text{row-0}} \\ \\ \boxed{\text{row-27}} \end{array} \end{array}$$

$$\boldsymbol{x} = \begin{array}{cc} \begin{bmatrix} A_{0,0} \\ A_{0,1} \\ A_{0,2} \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ A_{27,27} \end{bmatrix} & \begin{array}{l} \text{the first row} \\ \\ \\ \text{the second row} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \end{array}$$

$\boldsymbol{x} \in \mathbb{R}^{784}$

a vector ~an image ~ a data sample

Represent a customer by a vector

Each **row** is a feature vector of a customer

	A	B	C
1	income	spend	customer_ID
2	233	150	1
3	250	187	2
4	204	172	3
5	236	178	4
6	354	163	5
7	192	148	6
8	294	153	7
9	263	173	8
10	199	162	9

$$x = \begin{bmatrix} ID \\ income \\ spend \end{bmatrix}$$

In many applications, *customer ID* is not useful, so we remove it

$$x = \begin{bmatrix} income \\ spend \end{bmatrix}$$

x_1 : the 1st customer (1st row in the table)

x_2 : the 2nd customer (2nd row in the table)

Vector Distance Measure

- $\boldsymbol{x} = \begin{bmatrix} 0.1 \\ 1.2 \end{bmatrix}$, $\boldsymbol{y} = \begin{bmatrix} \mathbf{0.2} \\ \mathbf{2.1} \end{bmatrix}$

  the distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ is $\sqrt{(0.1 - \mathbf{0.2})^2 + (1.2 - \mathbf{2.1})^2}$

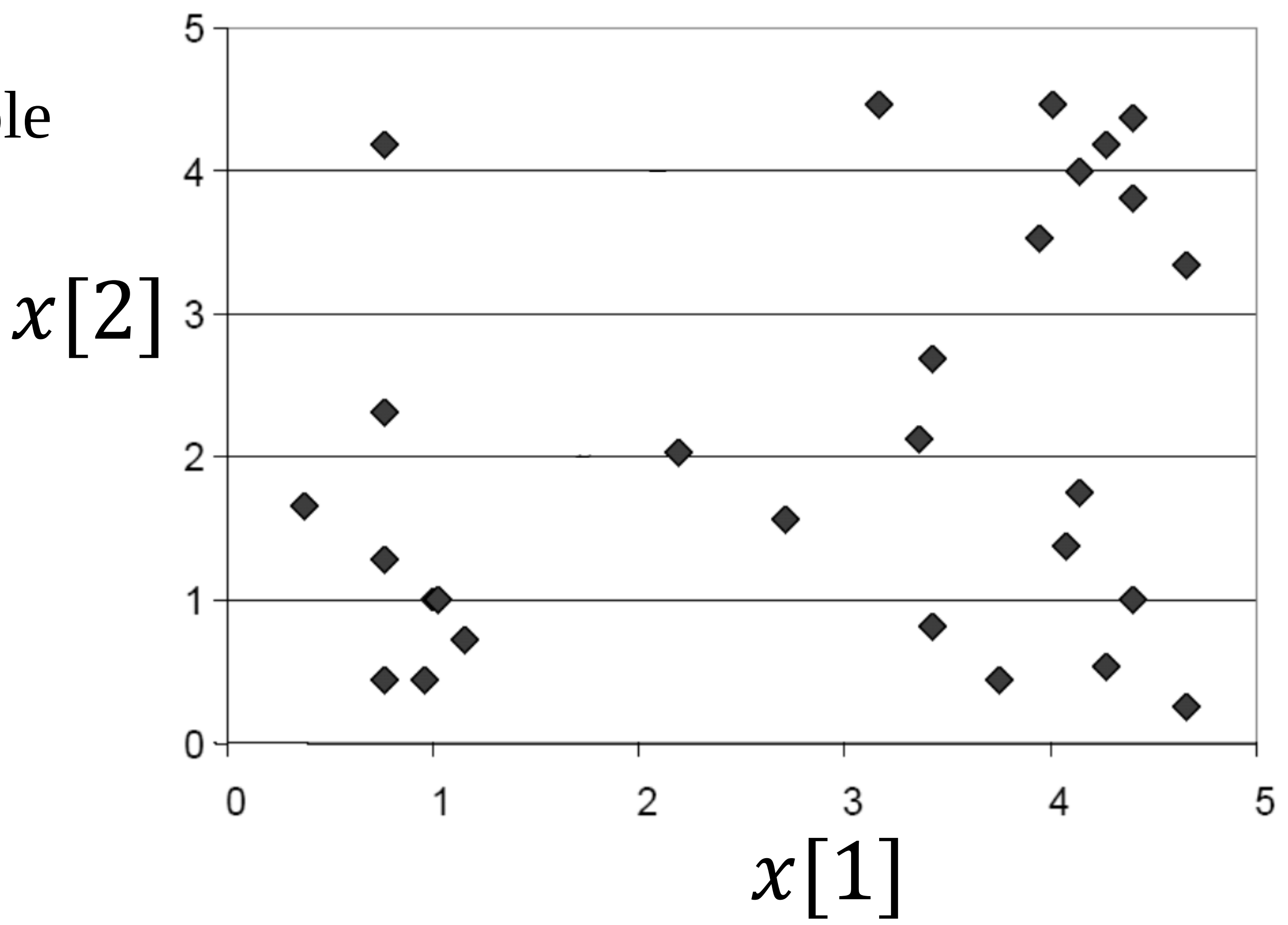
Run kmeans_cust_seg.ipynb

Before clustering, a dataset of vectors/samples

a feature vector $\boldsymbol{x}$ is a data sample

$$\boldsymbol{x} = \begin{bmatrix} x[1] \\ x[2] \end{bmatrix}$$

a data sample is also called a data point, i.e. a point in a *high* dimensional space



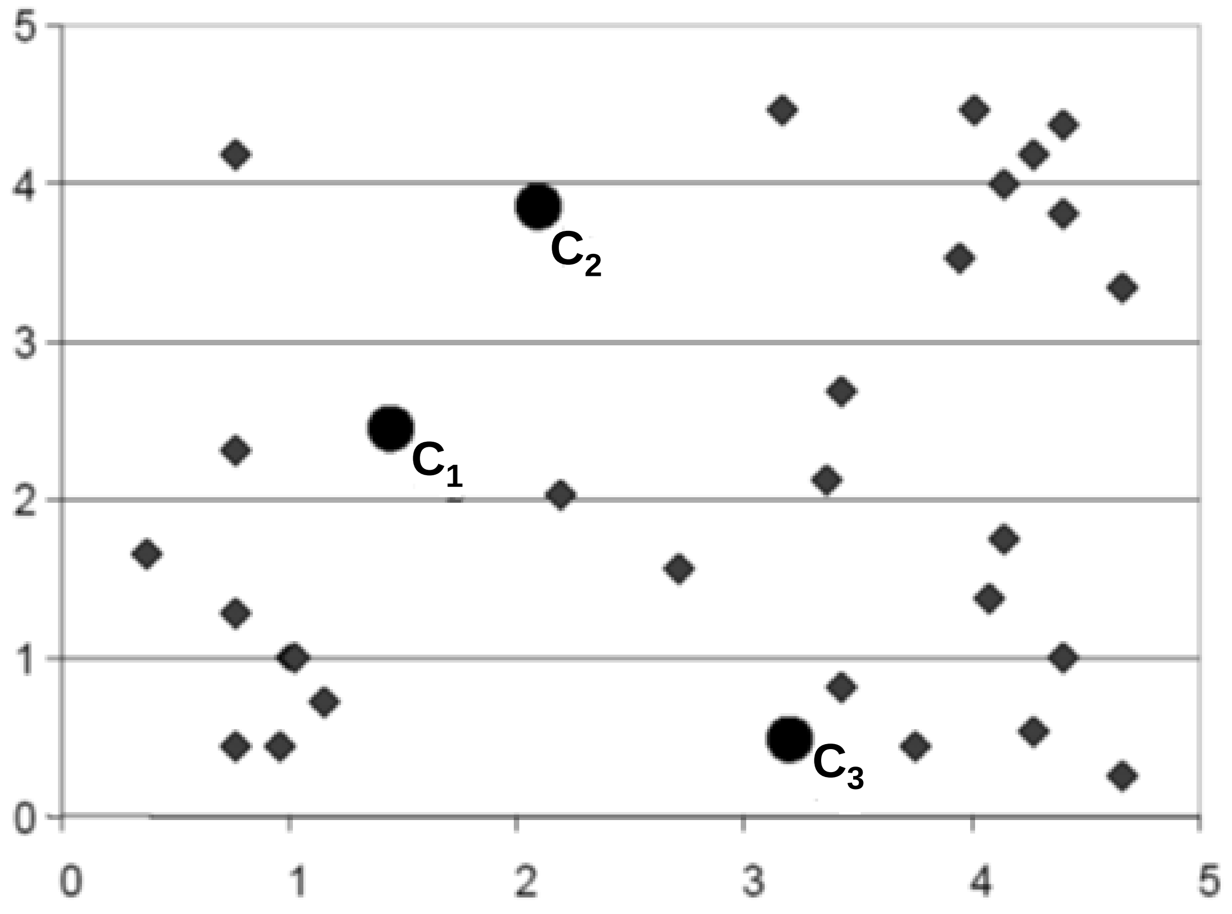
Apply k-means algorithm: Initialization

Initialization:

(1) The user (you) sets the number of clusters e.g., 3

(2) The algorithm will randomly initialize the cluster centers/centroids.

A cluster center is a vector. We get three random centers



c_1, c_2, c_3 are initial cluster centers at three random locations

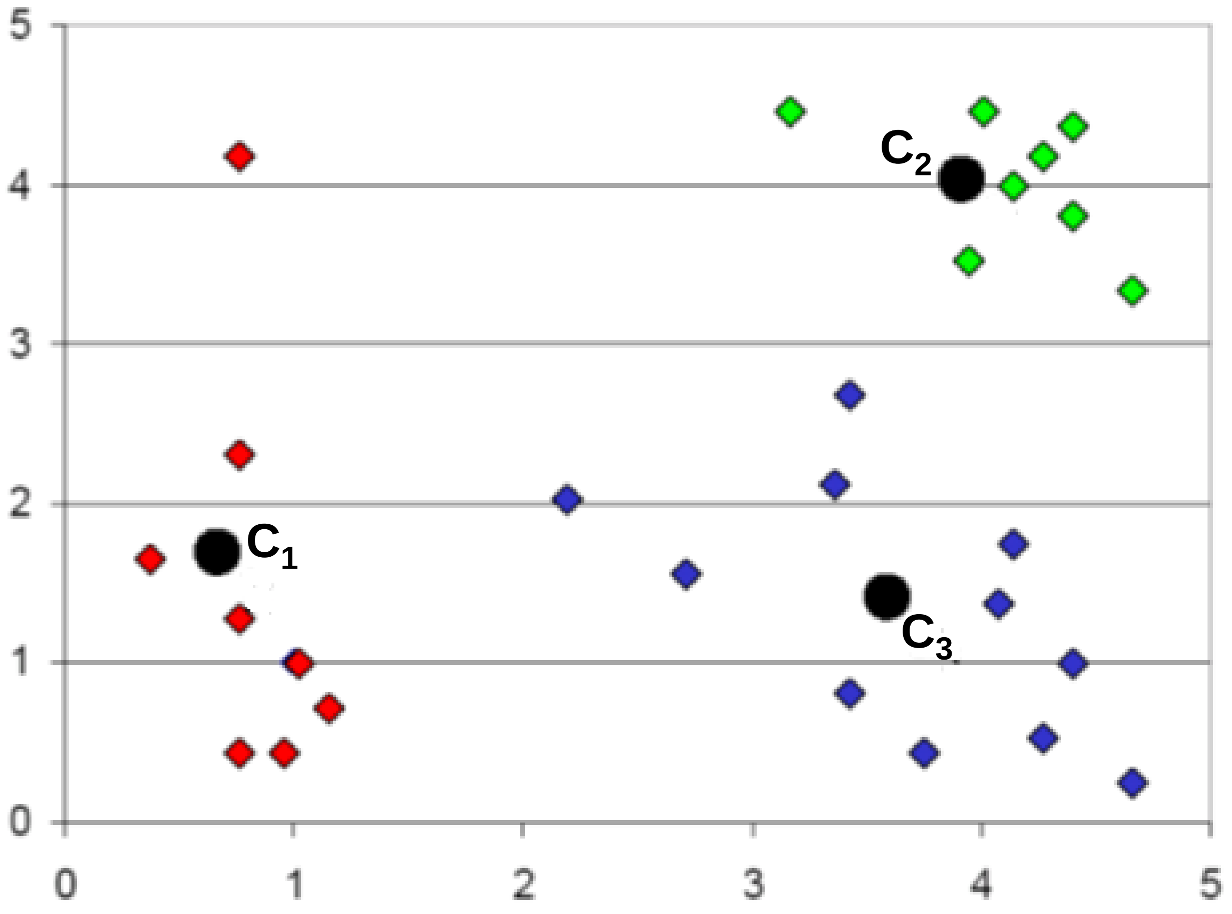
After k-means clustering, clusters/groups are formed

After k-means clustering:

- The data points are assigned to the three clusters
  red-cluster
  green-cluster
  blue-cluster

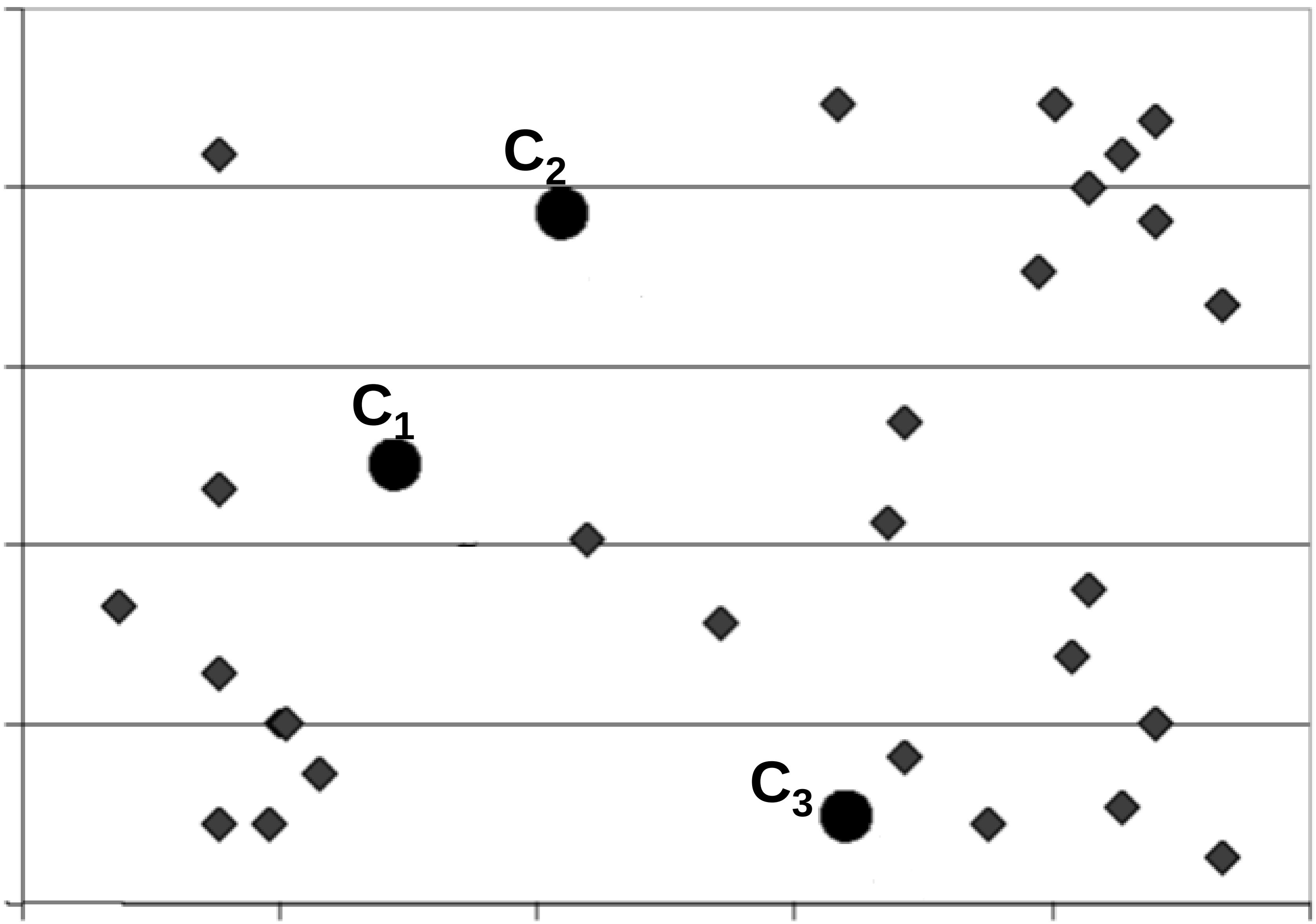
- Every data point has a cluster label that could be 1, 2, or 3

- The final cluster centers are different from the initial centers

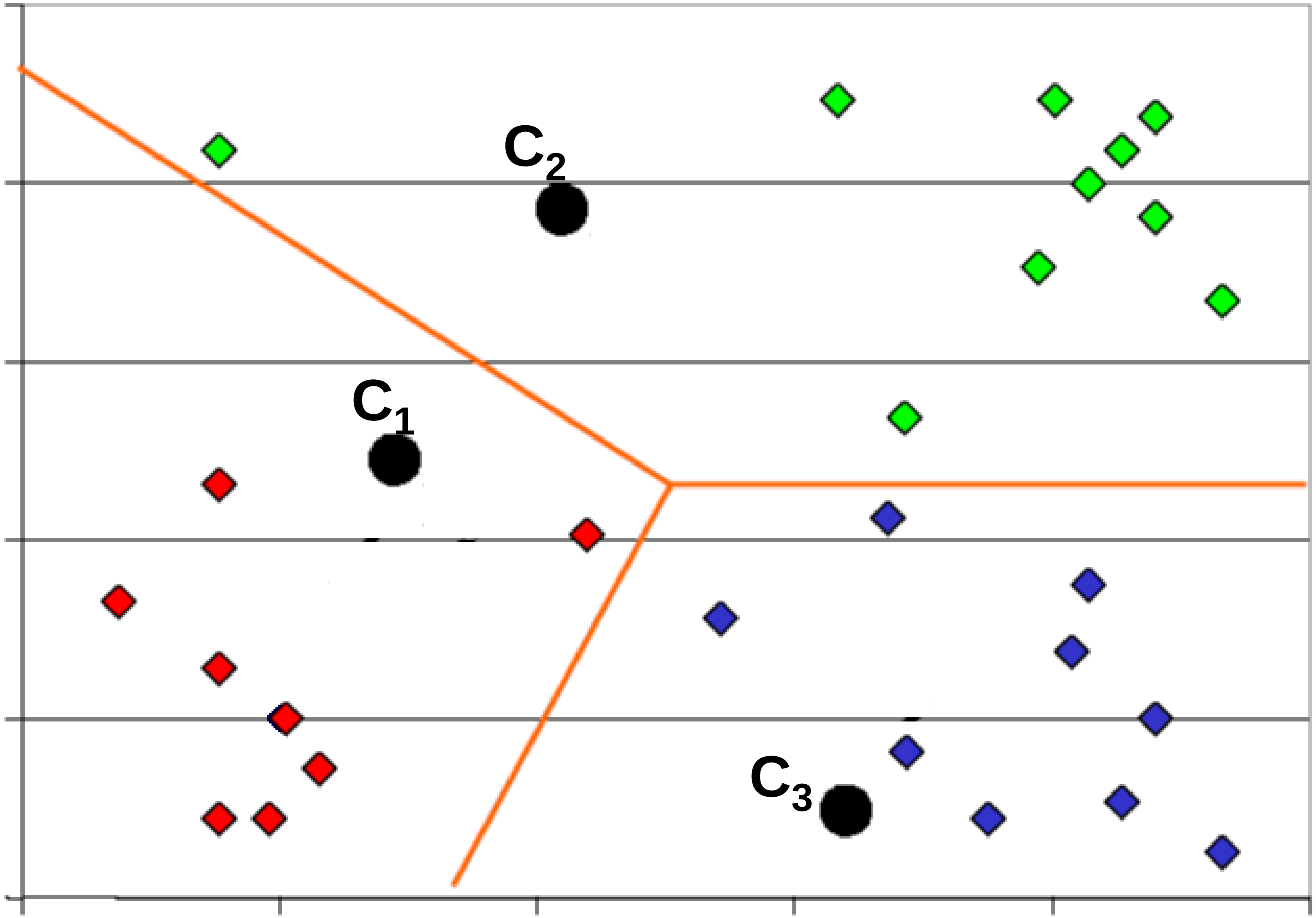


c_1, c_2, c_3 are the cluster centers

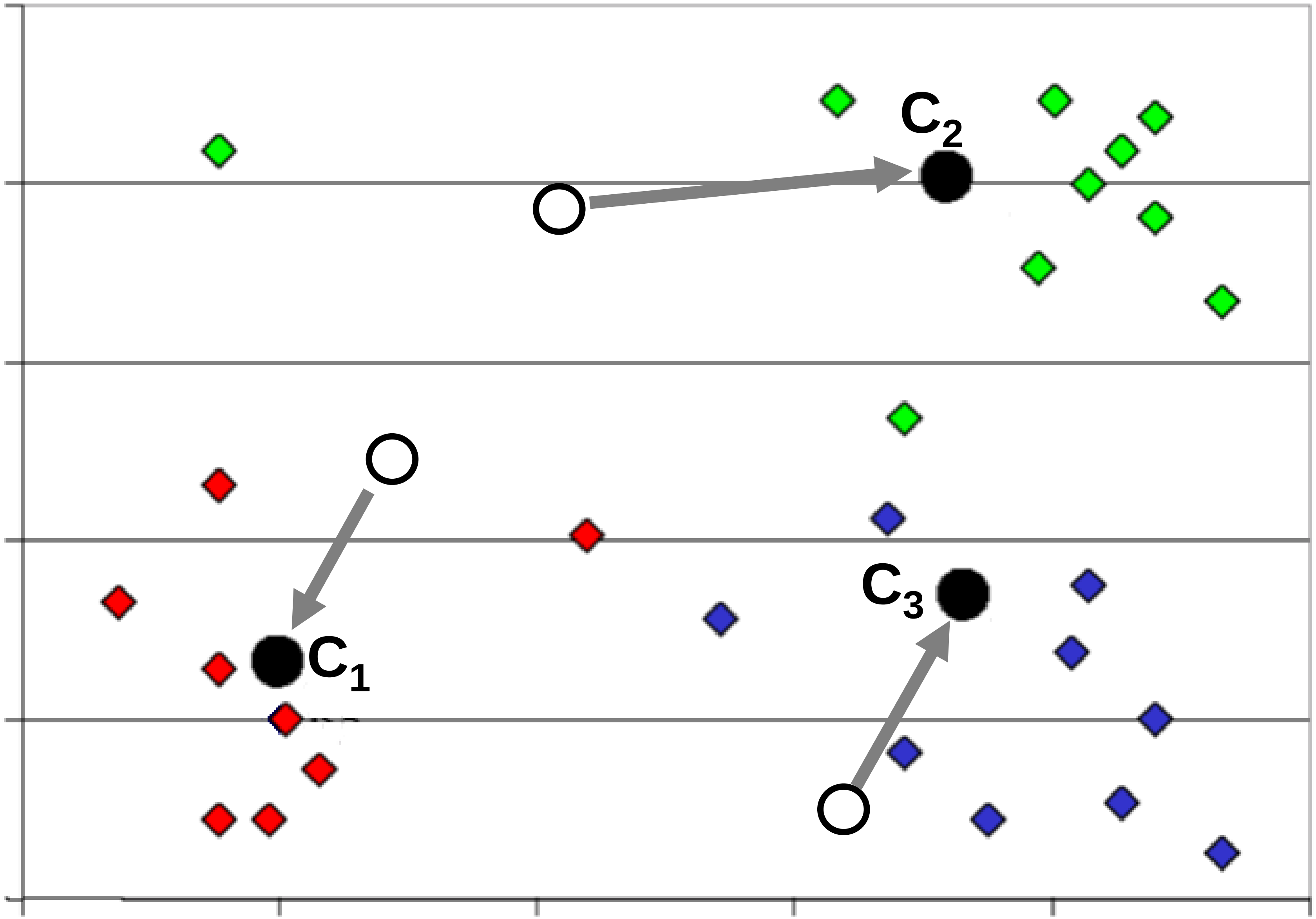
Initialization: the number of clusters and random locations of the cluster centers



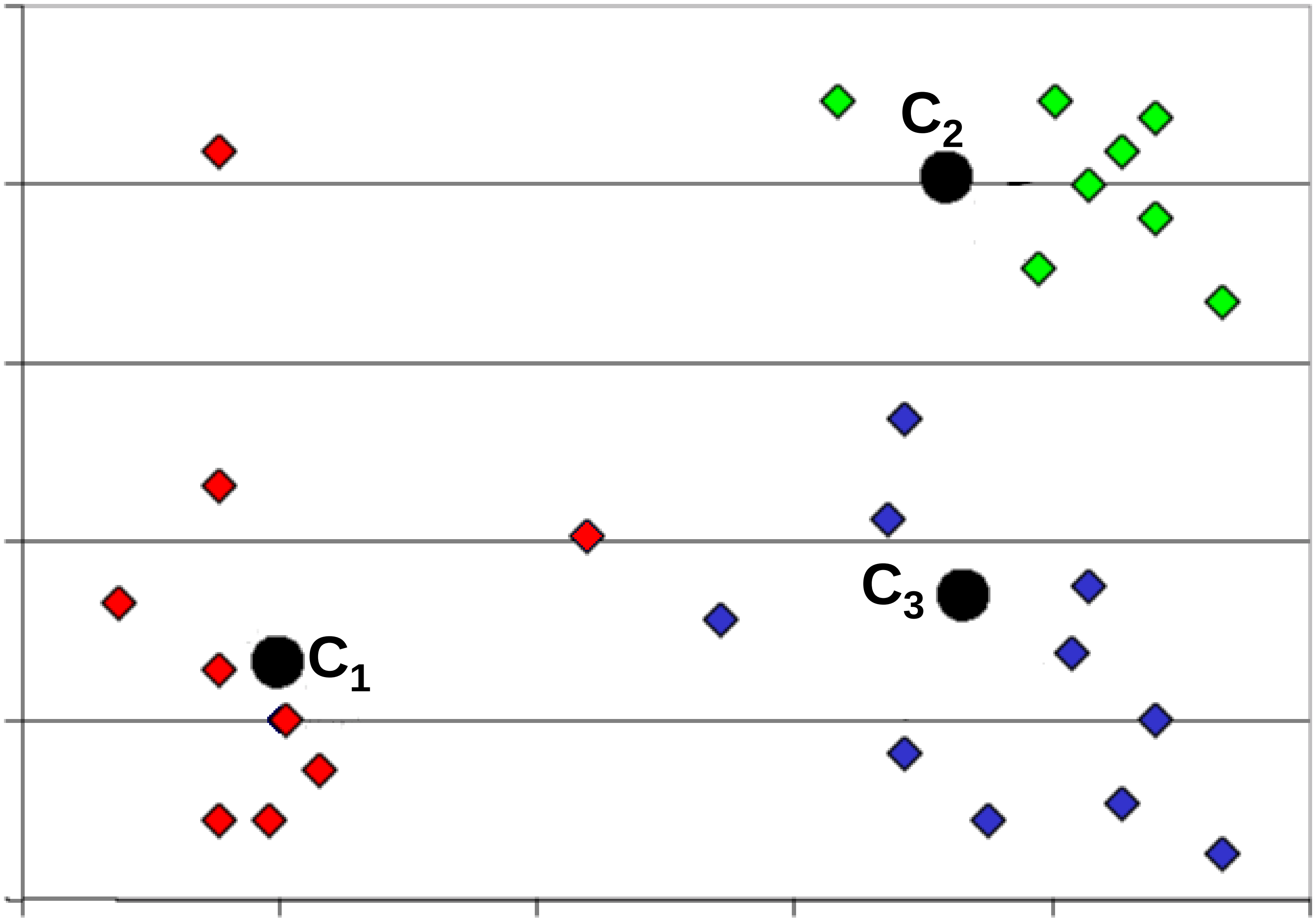
Assign Labels: assign each data point to the nearest cluster center



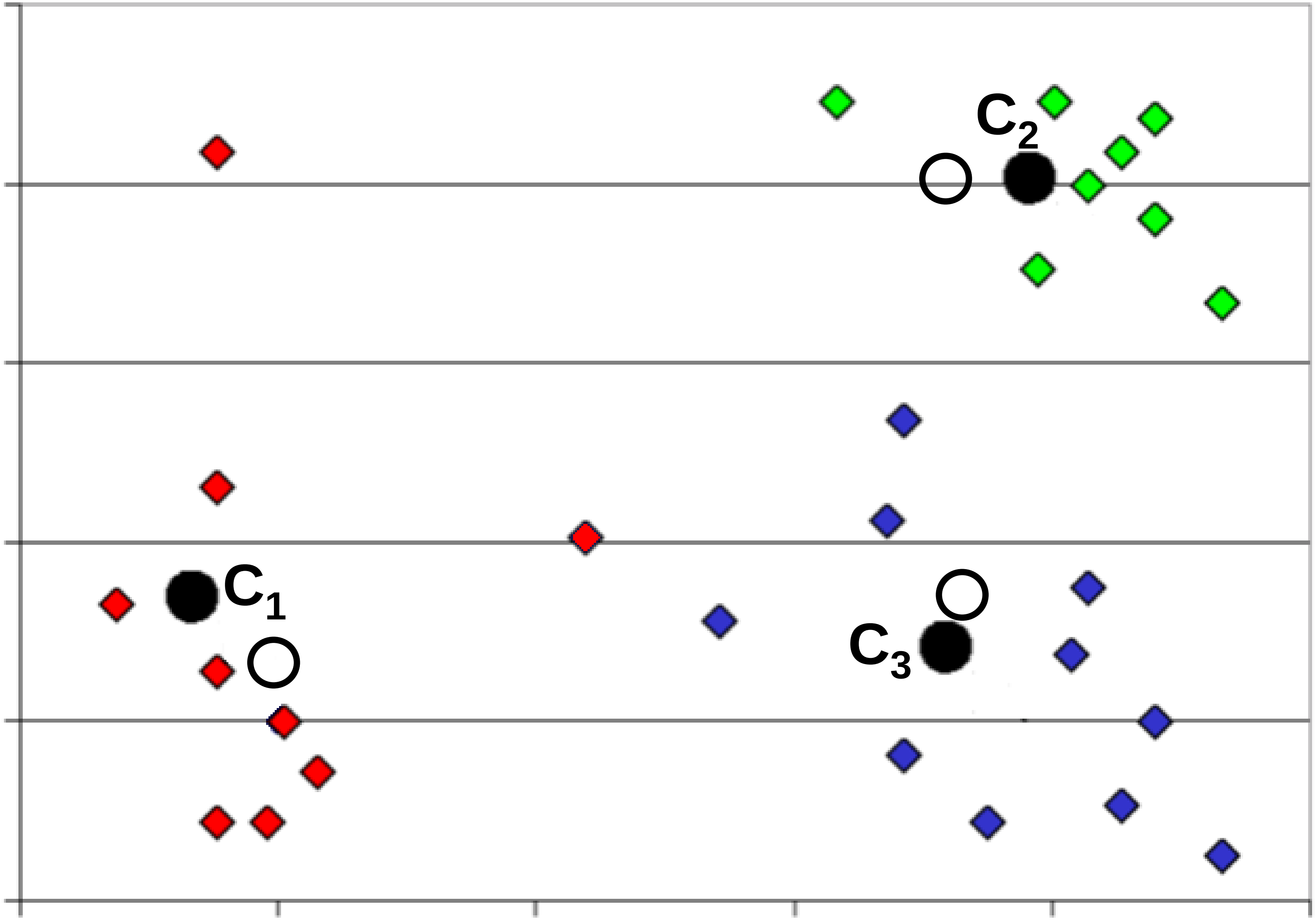
Update Centers: re-compute the center of each cluster



Assign Labels: assign each data point to the nearest cluster center



Update Centers: re-compute the center of each cluster



two steps run iteratively in the k-means algorithm

- Update Centers

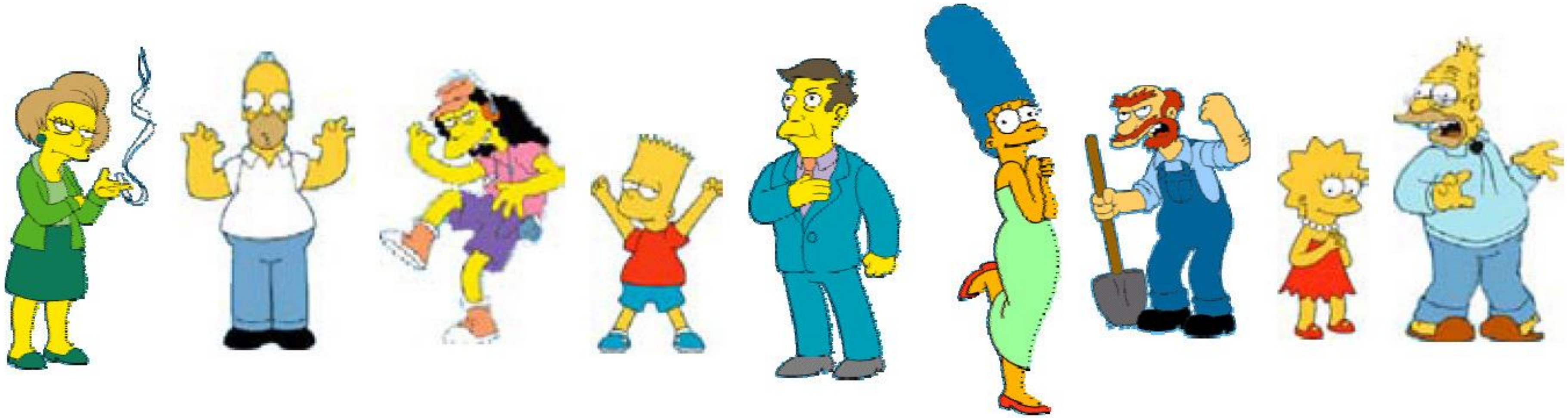
  for each cluster, move the center vector C to the average location of the data points in the cluster

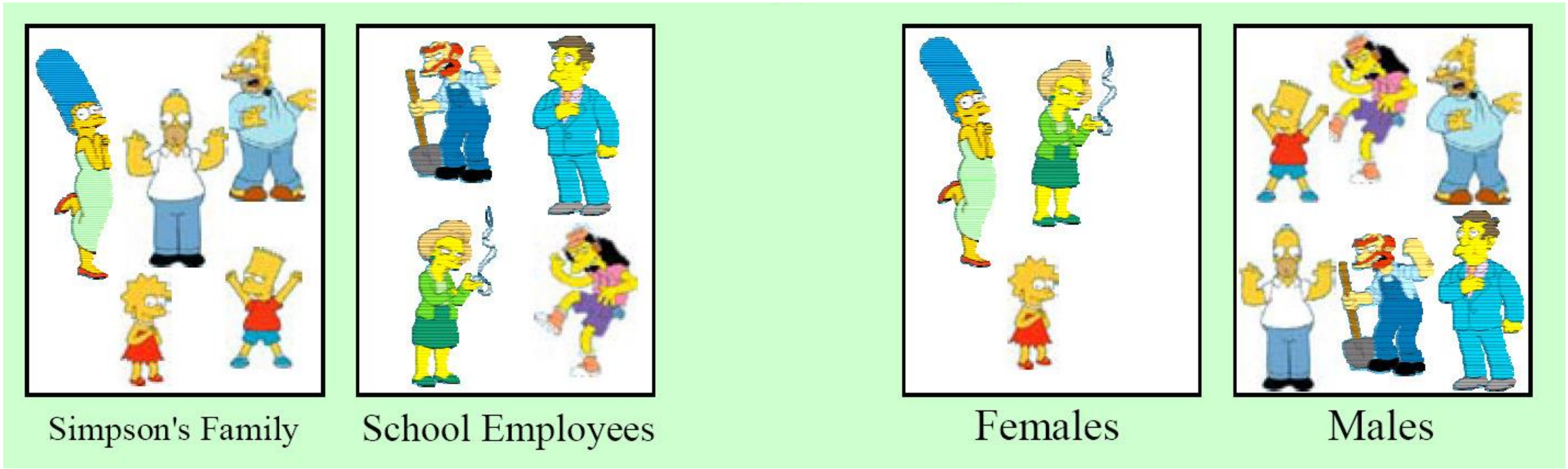
- Update Labels

  for each data point, find the nearest cluster center and then attach a cluster label to the data point

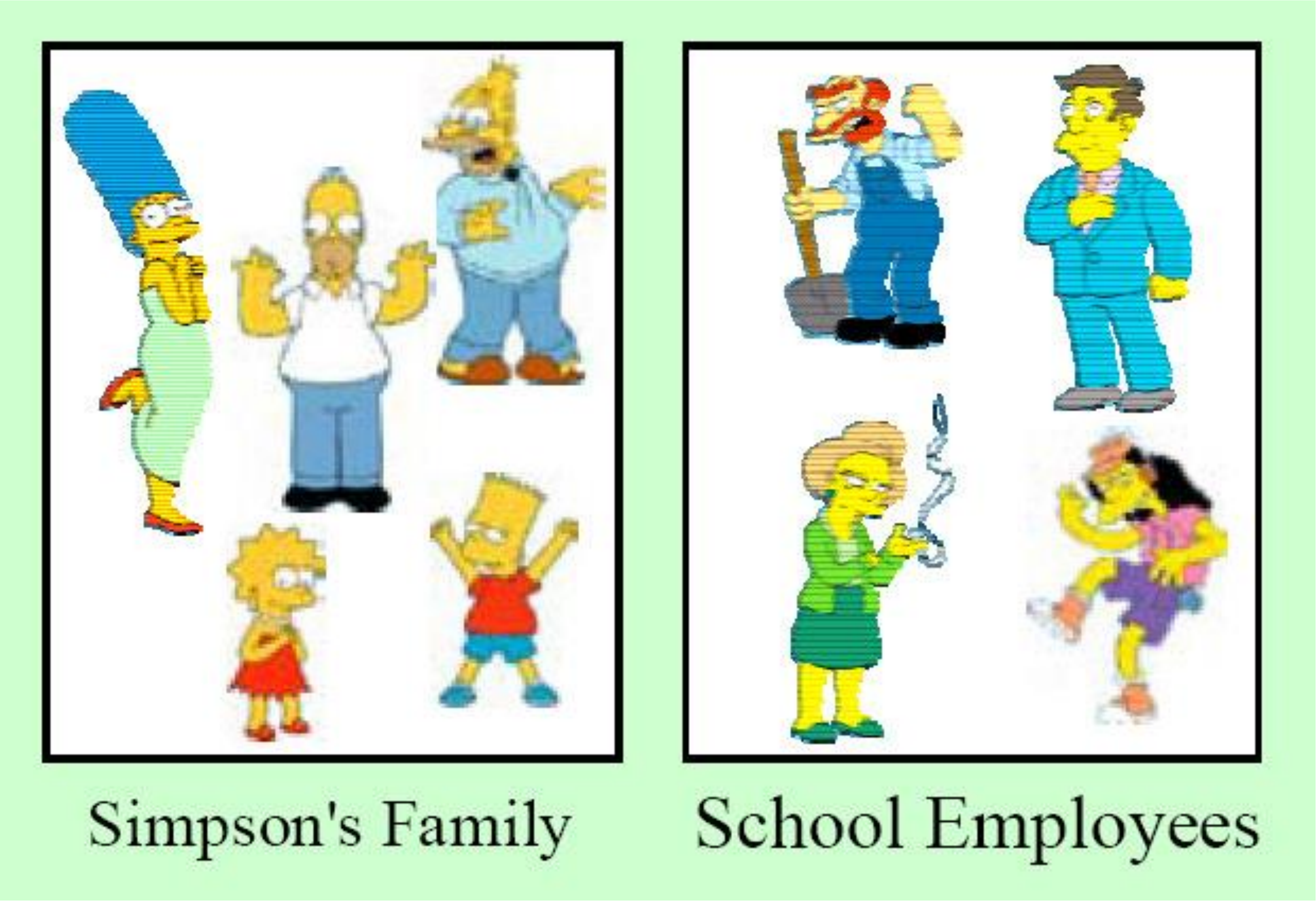
Let's implement K-means from scratch

Clustering is based on distance measure and feature vector





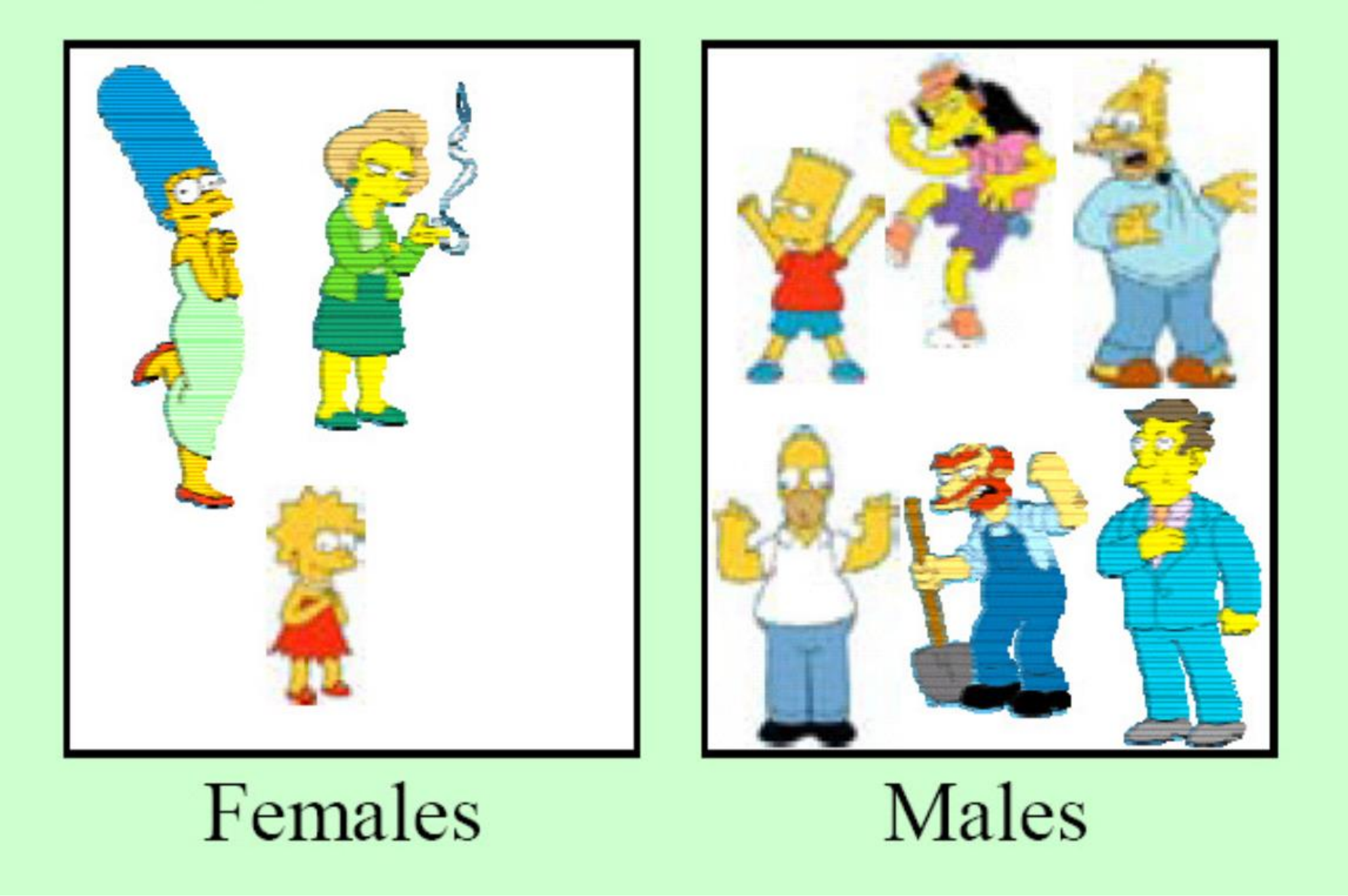
Clustering is based on distance measure and feature vector



Feature Vector

$$\boldsymbol{x} = [last\_name]$$

Clustering is based on distance measure and feature vector



Feature Vector

$$\boldsymbol{x} = [gender]$$

many distance/dissimilarity measures

https://www.psychologytoday.com/us/blog/canine-corner/201308/do-dogs-look-their-owners

So what is clustering in general?

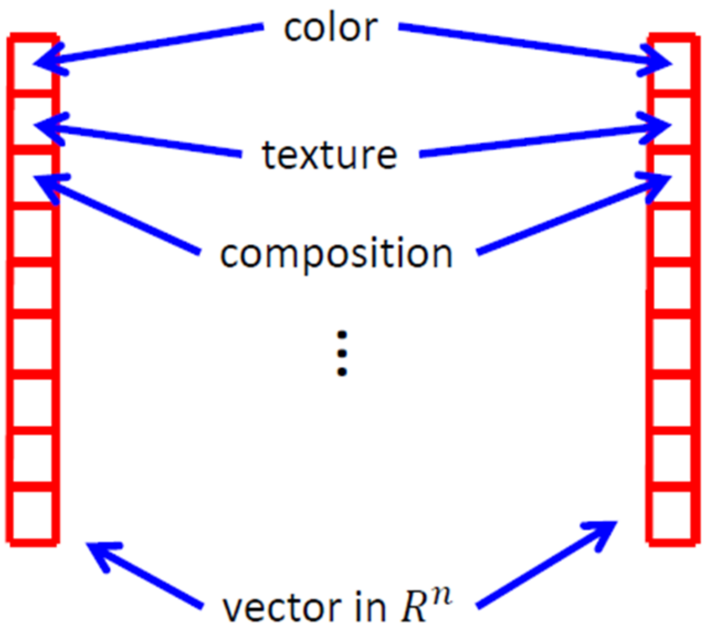
- You choose a distance/dissimilarity function
- The algorithm figures out the grouping of objects based on the distance function: distance(vectorA, vectorB)
- Data points within a cluster are similar
- Data points across clusters are not so similar

Feature Extraction Before Clustering

- Images of different sizes

Can not directly compare the two images
because they have different number of pixels
resize the images, or extract some features

small image



Objects in real life