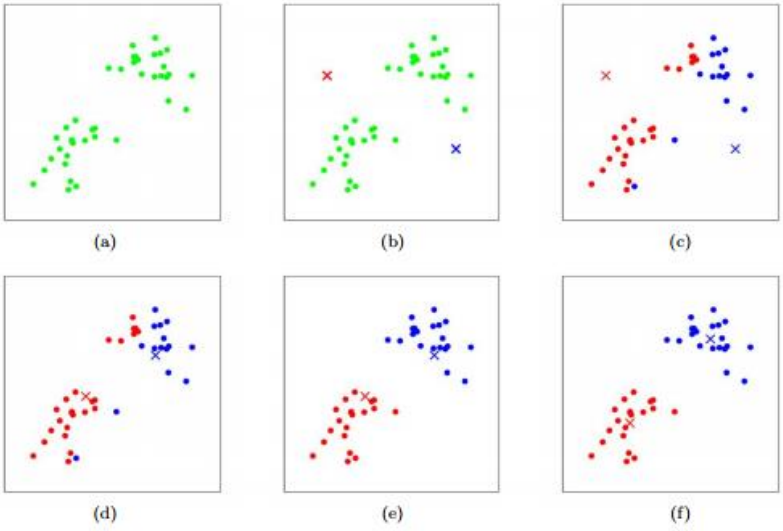
8.1 K-means Algorithm

8.1.1 What's K-means Algorithm?

The K-means algorithm is **an unsupervised algorithm**. The key idea it that clustering is carried out with k centroid points in the space, the objects closest to them are classified together, and the values of each clustering center are updated by iteration until the best clustering results are obtained.

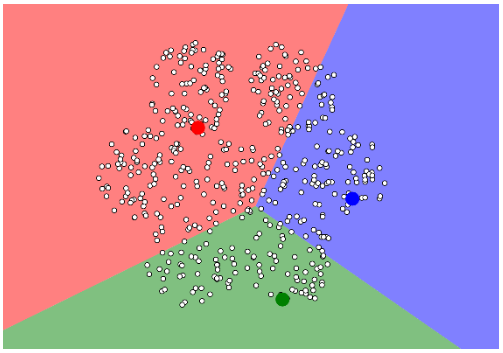
8.1.2 Process for K-means Algorithm

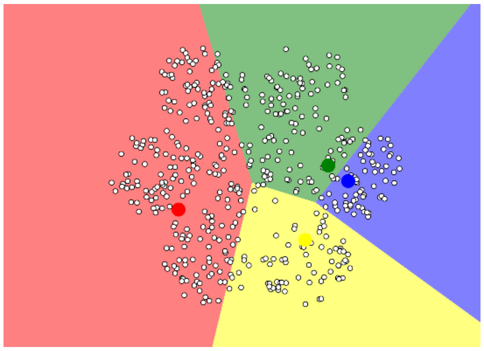
1) Select the k initial centers at random, actually it's better to select these k centers appropriately.
2) Find the distance between any sample and each center and classify the sample into the center with the shortest distance.
3) Update the central value of each cluster with means of all sample in every cluster.
4) For all k clustering centers, the iteration will end if the center value remains unchanged after the iterative method of (2) and (3), otherwise continue to iterate.



8.1.3 Parameters in K-means

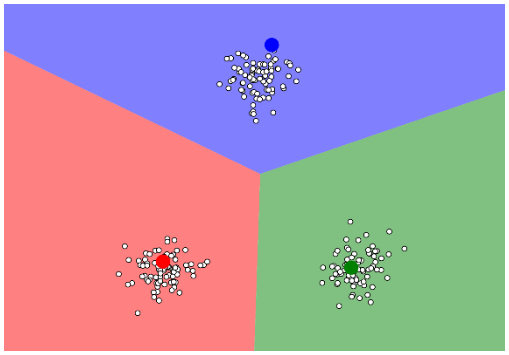
- **K:** decide how many clusters we need to divide.

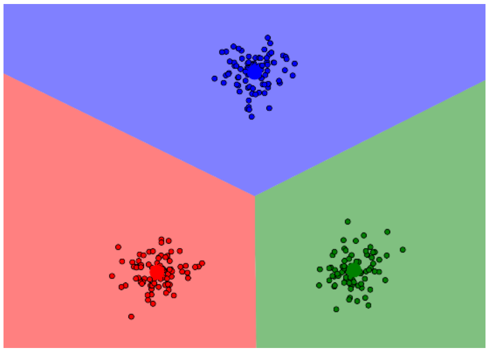




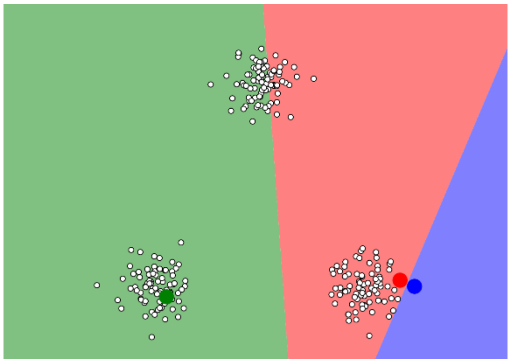
- **Centroid:** this is very important. Generally different centroids can produce the same results, but it can not be always. Please check out the following situations:

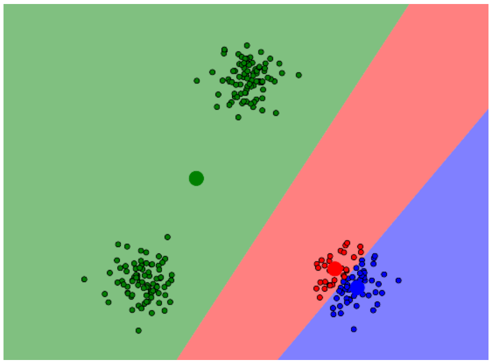
Situation 1:





Situation 2:





Therefore, we know different centroids maybe produce different results, so it’s better to do multiple experiment and get the mean of all of them. You can also find something new on the website: https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

- **Distance:** we basically use Euclidean Distance.

$$D = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{1}$$

- **Estimation:** we can use **Silhouette Coefficient** to estimate the model.
  - The sample clustering would be better if s(i) is close to 1.
  - The sample clustering would if it is close to -1.
  - The sample is on the boundary of clustering if s(i) is close to 0.

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} = \begin{cases} 1 - \frac{a(i)}{b(i)}, & a(i) < b(i) \\ 0, & a(i) = b(i) \\ \frac{a(i)}{b(i)} - 1, & a(i) > b(i) \end{cases} \tag{2}$$

  - a(i): no similarity in the cluster, which is the average distance from sample i to other samples in the same cluster.
  - b(i): the no similarity among clusters, which is the min{b_{i1}, b_{i2}, …, b_{ik}}, b_{ij} is the average distance from sample i to other samples in the different cluster.

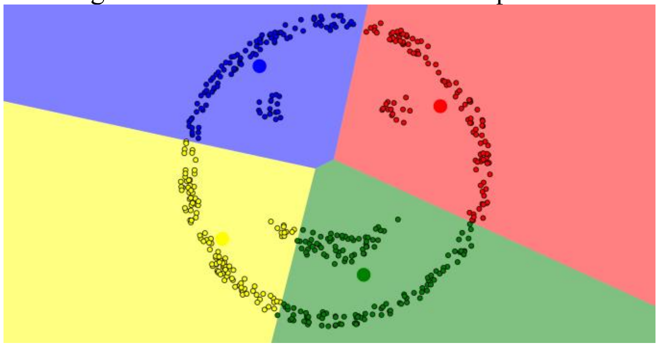
8.1.4 Pros and Cons

Pros:

- General algorithm
- Easy to understand every step
- Clustering effect is good

Cons:

- K is hard to estimate in advance
- Initializing the different centroids maybe produce different results
- It’s not good for some data with a circle shape.



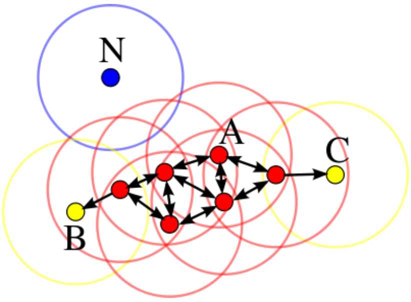
8.2 DBSCAN Algorithm

8.2.1 What’s DBSCAN Algorithm?

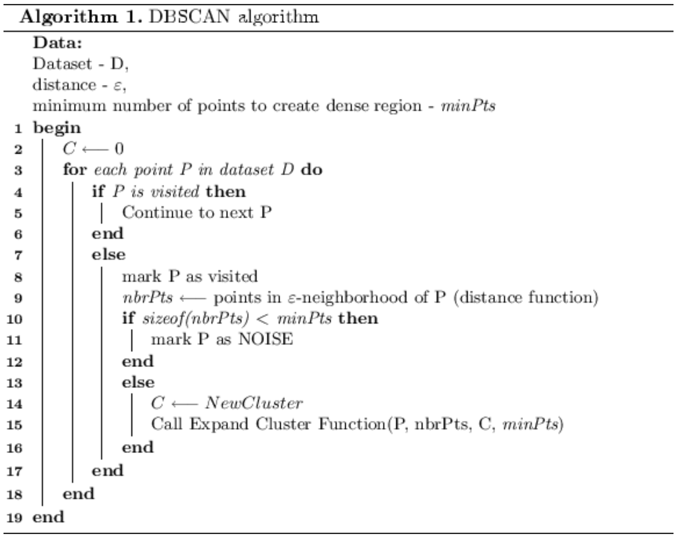
Density-Based Spatial Clustering of Applications with Noise (DBSCAN) Algorithm is another unsupervised algorithm, which is used to cluster the data with a circle shape. Because **the core idea of this algorithm is to cluster based on the density of points**. Of course, it can also work well for the typical clustering which K-means can be used with.

Basic Concepts:

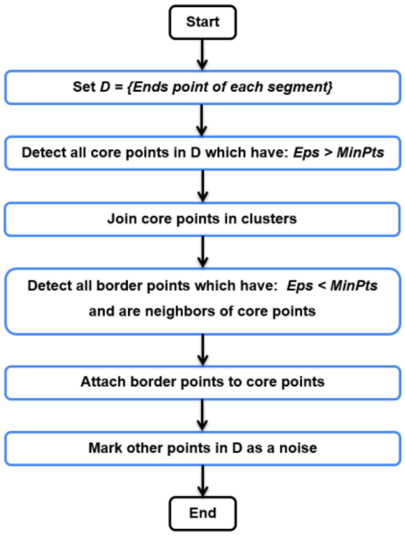
- ϵ - neighborhood: a circle with the radius r.
- Core point: there are at least some points (count >= threshold: MinPoints) in the ϵ - neighborhood. For example, red points.
- Boundary point: they are in the ϵ - neighborhood of the most outside circle. For example, yellow points.
- Outlier: the most outside points which can not be in any other circle beside itself. For example, blue points.



8.2.2 Process for K-means Algorithm



```
Algorithm 1. DBSCAN algorithm
Data:
Dataset - D,
distance - $\varepsilon$,
minimum number of points to create dense region - $minPts$
begin
    $C \longleftarrow 0$
    for each point P in dataset D do
        if P is visited then
            Continue to next P
        end
        else
            mark P as visited
            $nbrPts \longleftarrow$ points in $\varepsilon$-neighborhood of P (distance function)
            if $sizeof(nbrPts) < minPts$ then
                mark P as NOISE
            end
            else
                $C \longleftarrow NewCluster$
                Call Expand Cluster Function(P, nbrPts, C, $minPts$)
            end
        end
    end
end
```



Start

Set *D = {Ends point of each segment}*

Detect all core points in D which have: *Eps > MinPts*

Join core points in clusters

Detect all border points which have: *Eps < MinPts* and are neighbors of core points

Attach border points to core points

Mark other points in D as a noise

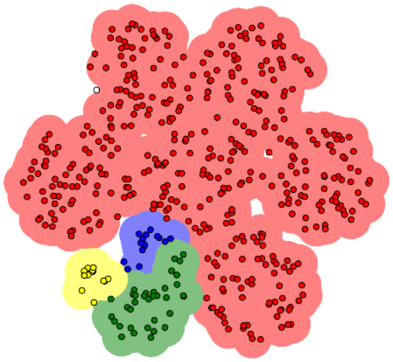
End

8.2.3 Parameters in DBSCAN Algorithm

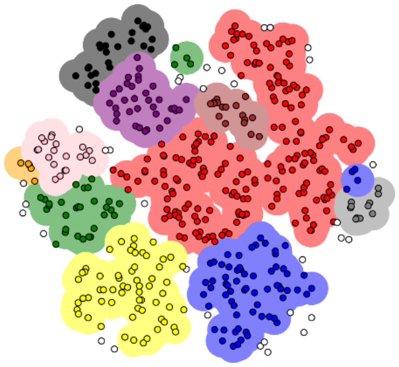
Radius r: it is a very important parameter influencing the size of circle (the points included).

- Bigger the r is, the more the points included in the circle are, the less categories are, the less outlier are.
- Smaller the r is, the less the points included in the circle are, the more categories are, the more outlier are.

R = 1.0, minPoints = 4



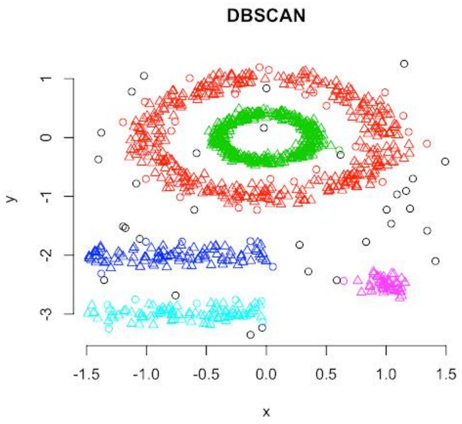
R = 0.8, minPoints = 4

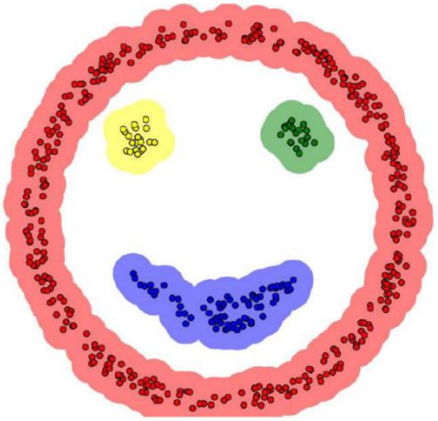


8.2.4 Pros and Cons

Pros:

- It can be used for data with any shape including the data with a circle.
- It does not need to input how many clusters in advance.
- It is good for detect outliers.





Cons:

- It will be bad when the data density is not uniform.
- It is not easy to choose the radius r.