Idea: unbalanced data (ideal output has very different number of samples in it)

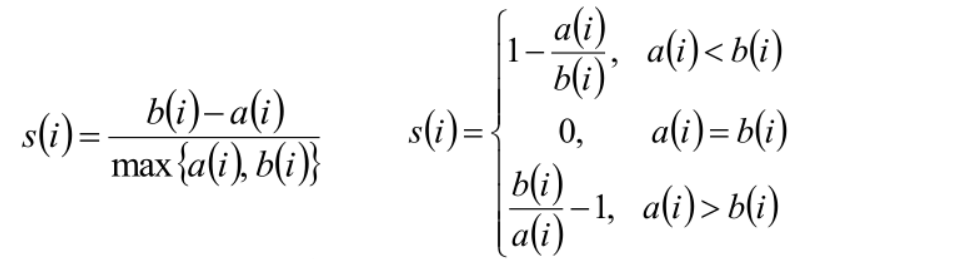
**1. estimation:**

**1.1 silhouette\_score**

ai: average distance from point i to all points in the same cluster with i

bij：average distance from point i to all points from cluster Cj (other than the cluster that i belongs to)

bi =min{bi1, bi2, ..., bik}



**2. models:**

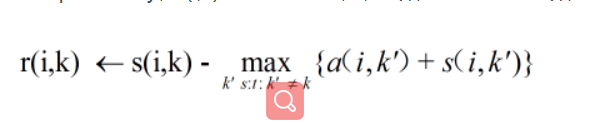
**2.1 AP(affinitypropagation)**

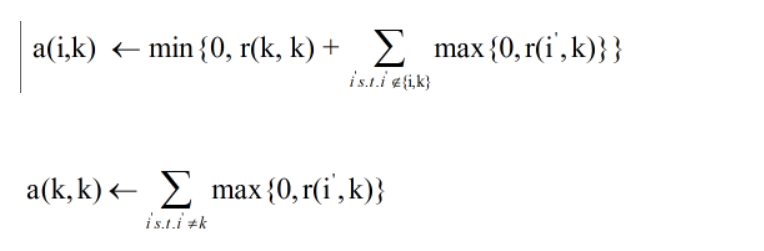
s(i,k): similarity matrix (the input )

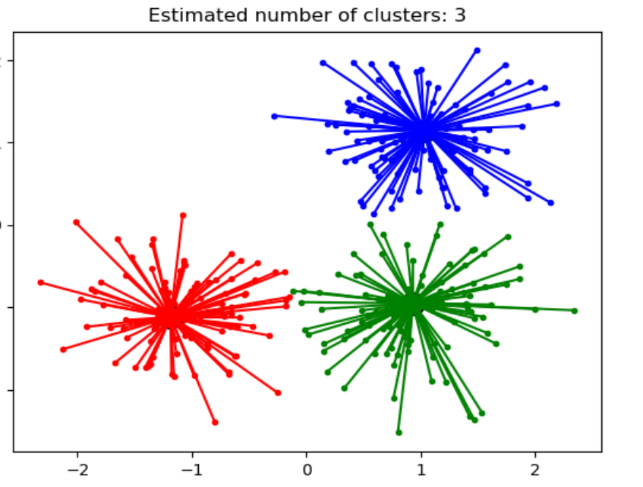
r(i,k): responsibility matrix

a(i,k): availability

Iteration (until converge):





Aim:

Additional notes: adjust S(k, k) to control the number of clusters you get!

**2.2 Agglomerative Clustering Algorithm**

1.define every points as a cluster

2.iterate：merge two points whose “distance” is the smallest

3.untill there are only one cluster

Different ways of estimating distance between two clusters:

Single-linkage: min(A,B)

Complete-linkage: max(A,B)

Group average: ave(A,B)

Ward: minimize the sum of squared differences within all clusters (result should be similar to k means)

Flaw: greedy

**2.2.1 wards algorithm:**

During the process of clustering:

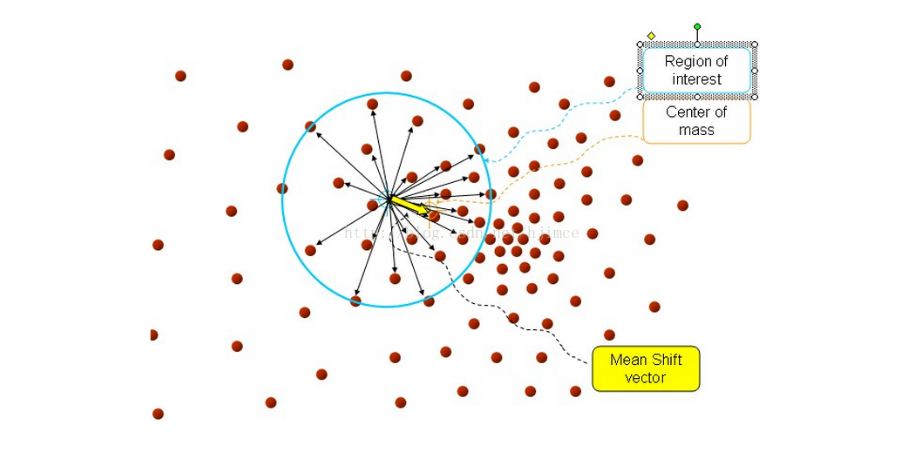
minimizes the sum of squared differences within all clusters in the next step.

2.3 mean shift

Iterate until every point is labelled:

Choose one center from all points

Move that center to the direction of the most dense area until converge:



Label it as one cluster (all the points in every circle in the iteration should be in this cluster)

If the new center is too close to an existing center, merge those two centers

label all points in the cluster