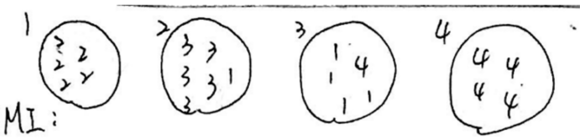
CS 145 HW3

ID: 004943505
Name: Calvin Chen

1.
Purity = 1/20*(5+5+4+4) = 0.9



NMI:

$P(C_1 \cap W_2) = \frac{5}{20}$ $\quad P(C_1) = \frac{5}{20}$ $\quad P(W_1) = \frac{5}{20}$

$P(C_2 \cap W_3) = \frac{5}{20}$ $\quad P(C_2) = \frac{6}{20}$ $\quad P(W_2) = \frac{5}{20}$

$P(C_2 \cap W_1) = \frac{1}{20}$ $\quad P(C_3) = \frac{5}{20}$ $\quad P(W_3) = \frac{5}{20}$

$P(C_3 \cap W_1) = \frac{4}{20}$ $\quad P(C_4) = \frac{4}{20}$ $\quad P(W_4) = \frac{5}{20}$

$P(C_3 \cap W_4) = \frac{1}{20}$

$P(C_4 \cap W_4) = \frac{4}{20}$

$$I(\Omega, C) = \frac{5}{20}\log\left(\frac{5\times20}{5\times5}\right) + \frac{5}{20}\log\left(\frac{5\times20}{6\times5}\right) + \frac{1}{20}\log\left(\frac{1\times20}{6\times5}\right) + \frac{4}{20}\log\left(\frac{4\times20}{5\times5}\right) + \frac{1}{20}\log\left(\frac{1\times20}{5\times5}\right) + \frac{4}{20}\log\frac{4\times20}{4\times5}$$

$$= 1.126$$

$$H(\Omega) = -\left(\frac{5}{20}\log\frac{5}{20}\right)\times 4 = 1.386$$

$$H(C) = -\left(\frac{5}{20}\log\frac{5}{20}\times 2 + \frac{6}{20}\log\frac{6}{20} + \frac{4}{20}\log\frac{4}{20}\right) = 1.376$$

$$\Rightarrow NMI = \frac{1.126}{\sqrt{1.386\times1.376}} = 0.815$$

$$TP = \frac{5\times4}{2} + \frac{4\times3}{2} + \frac{4\times3}{2} + \frac{5\times4}{2} = 32$$

$$FP = 5+4 = 9$$

$$FN = 4+4 = 8$$

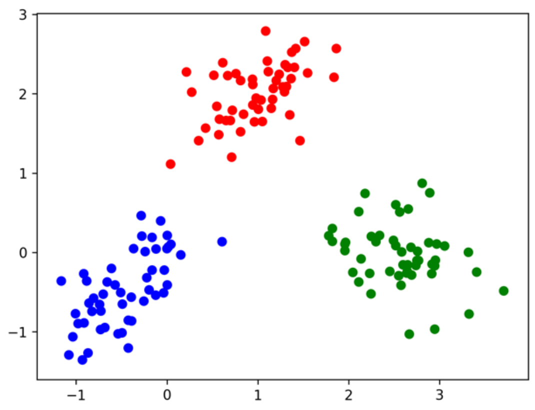
$$\text{Precision} = \frac{32}{41} = 0.78$$

$$\text{Recall} = \frac{32}{40} = 0.8$$

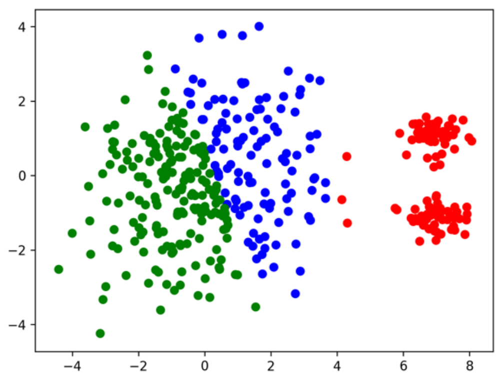
$$F\text{-measure} = \frac{2\times\frac{32}{41}\times0.8}{\frac{32}{41}+\frac{32}{40}} = 0.79$$

2.

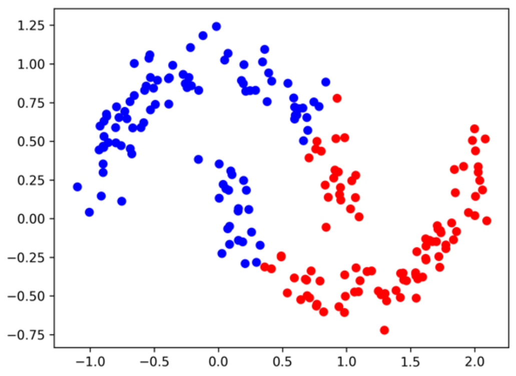
Dataset 1: Purity = 1.0, NMI = 1.0



Dataset 2: Purity = 0.8675, NMI = 0.464

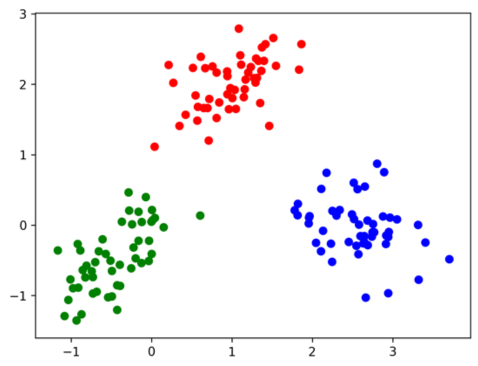


Dataset 3: Purity = 0.78, NMI = 0.170

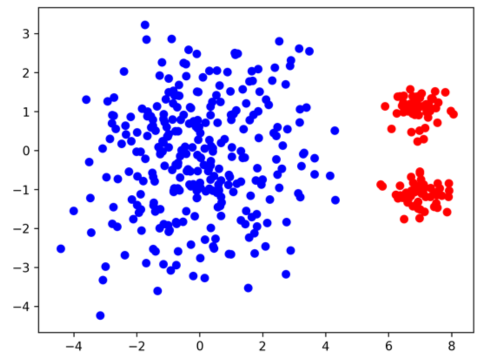


K-means algorithm is efficient, which runs O(t*k*n), where t = times of iteration, k = number of clusters, n = number of objects.
However, we need to specify k in advance and the algorithm is sensitive to noise. Also, it is not suitable to discover clusters like dataset3, which is a non-convex shape. Moreover, K-means only applicable to objects in a continuous n-dimension space.

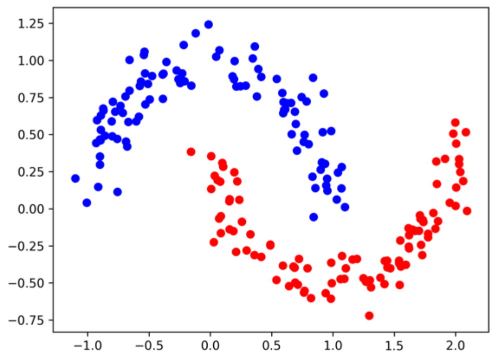
3.
Parameter 1: minPts = 3, epsilon = 0.24*sum_of_distance/(n^2)
Dataset 1: Purity = 1.0, NMI = 1.0



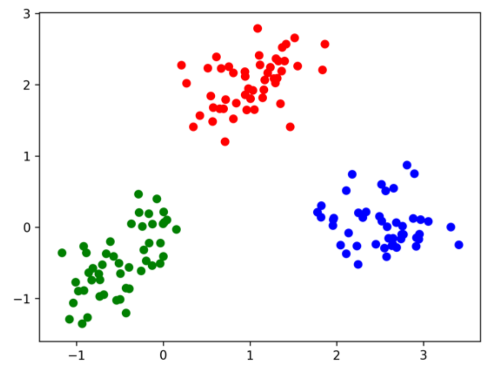
Dataset 2: Purity = 0.865, NMI = 0.688



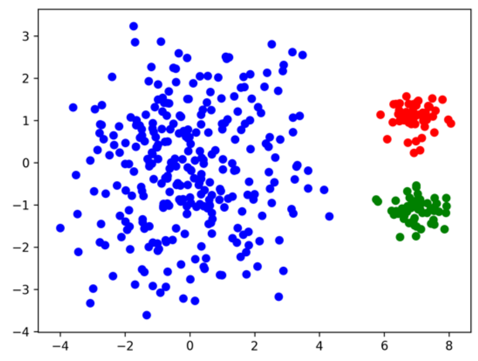
Dataset 3: Purity =1.0, NMI = 1.0



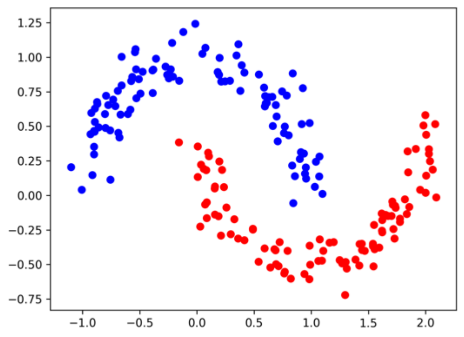
Parameter 2: minPts = 3, epsilon = 0.175*sum_of_distance/(n^2)
Dataset 1: Purity = 0.96, NMI = 1.0



Dataset 2: Purity = 0.98, NMI = 1.0

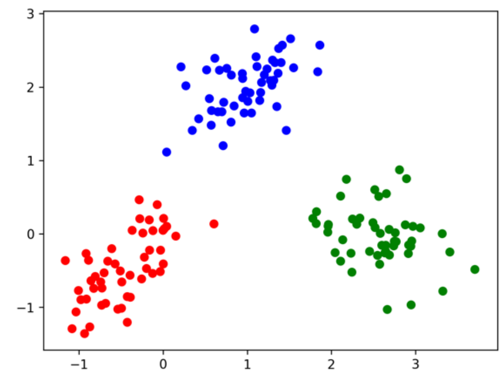


Dataset 3: Purity =1.0, NMI = 1.0

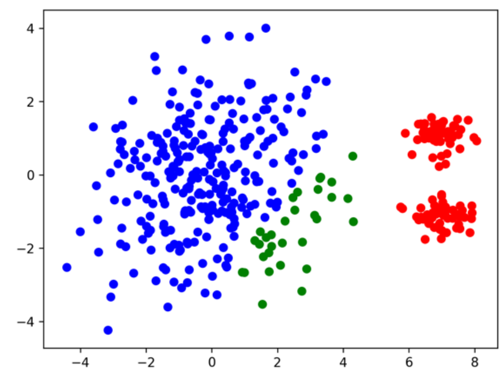


DBSCAN clusters data based on density. Thus, there is no need to specify number of clusters in advance. Also, it could handle noise and discover clusters of arbitrary shape like dataset 3. However, it need to determine density parameters such as minPts and epsilon as a termination condition. The two different case Parameter 1 and Parameter 2 show that the parameter defined can effect the result of clustering.

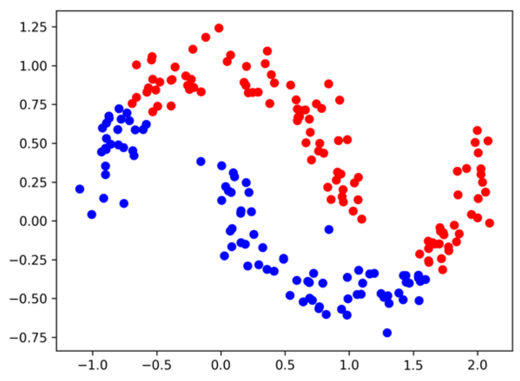
4.
Dataset 1: Purity = 1.0, NMI = 1.0



Dataset 2: Purity = 0.875, NMI = 0.609



Dataset 3: Purity =0.69, NMI = 0.0759



GMM is more general than partitioning different densities and sizes of clusters. Also, the results may satisfy the statistical assumption of the generative models. However, it can only deal with spherical clusters. Thus, is cannot perform well in dataset 3. Also, if the number of distribution is large, the computation would be expensive. And just like Kmeans, GMM is hard to estimate the number of clusters.