HW #4:

1.

Calculating Purity:

Output Label 1: 5 2's

Output Label 2: 5 3's 1 1

Output Label 3: 3 1's, 1 4, 1 1

Output Label 4: 4 4's.

Purity = (5+5+3+4) / 20 = .85

Precision, Recall, F-Measure

TP: 4+3+2+1+3+2+1+3+2+1+4+3+2+1 = 32

FP: 4 + 4 =8

FN: 5+4=9

Precision = TP/(TP+FP) = 32/40 = .8

Recall = TP/(TP+FN) = 32/41 = .78

F-measure = 2*Precision*Recall/(Precision + Recall) = .78987

NMI:

P(cluster 1) = ¼

P(cluster 2) = ¼

P(cluster 3) = ¼

P(cluster 4) = ¼

Used wolfram alpha to calculate H

H(Clusters) = 2

P(Class 1) = 1/4

P(Class 2) = 3/10

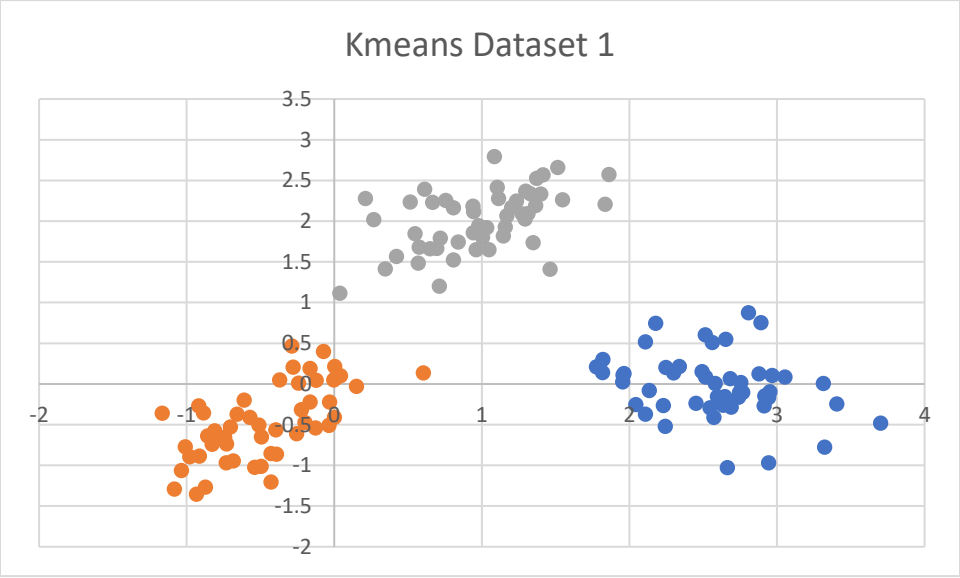
P(Class 3) = ¼

P(Class 4) = 1/5

H(Classes) = 1.9855

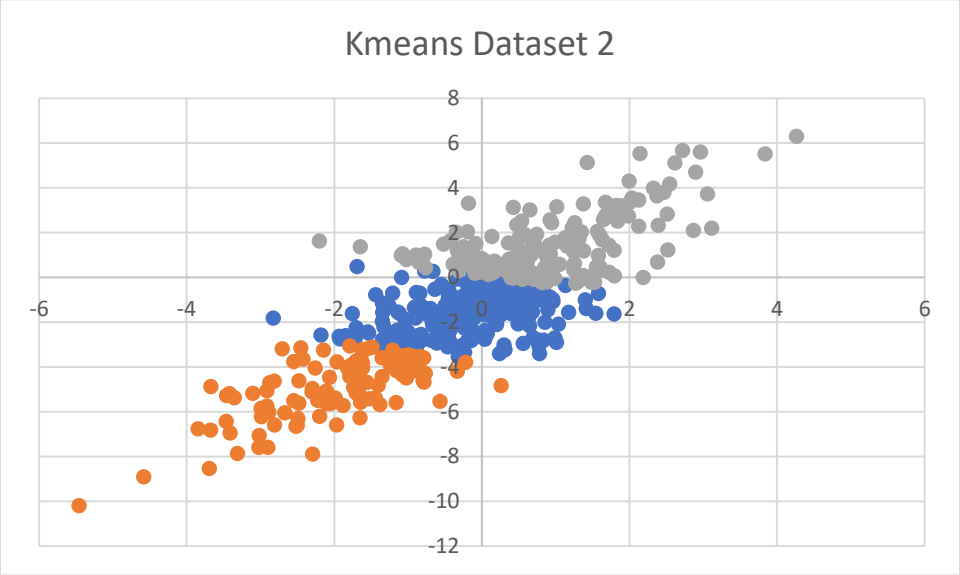
I = 5/20*log(20*5/16) + 4/20*log(20*4/16) + 1/20*log(20*1/16) +1/20*log(20*1/16) +4/20*log(20*4/16) + 5/20*log(20*5/16) = 2.2829

NMI = I/sqrt(H(Cluster) * H(Class)) = 1.145



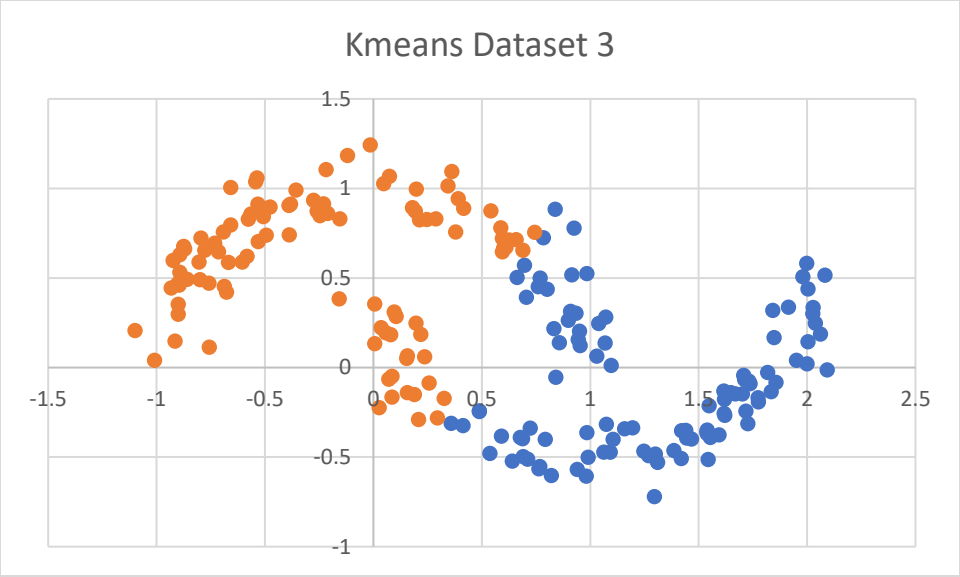

Purity: 1.0

NMI: 1.0




Purity: .764

NMI: 0.04685

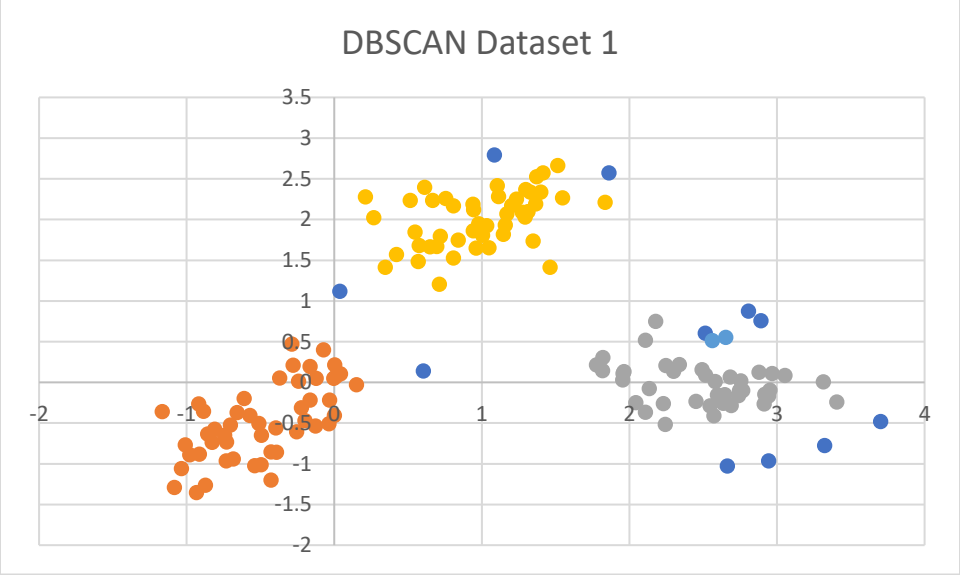



Purity: 0.76

NMI: .145025

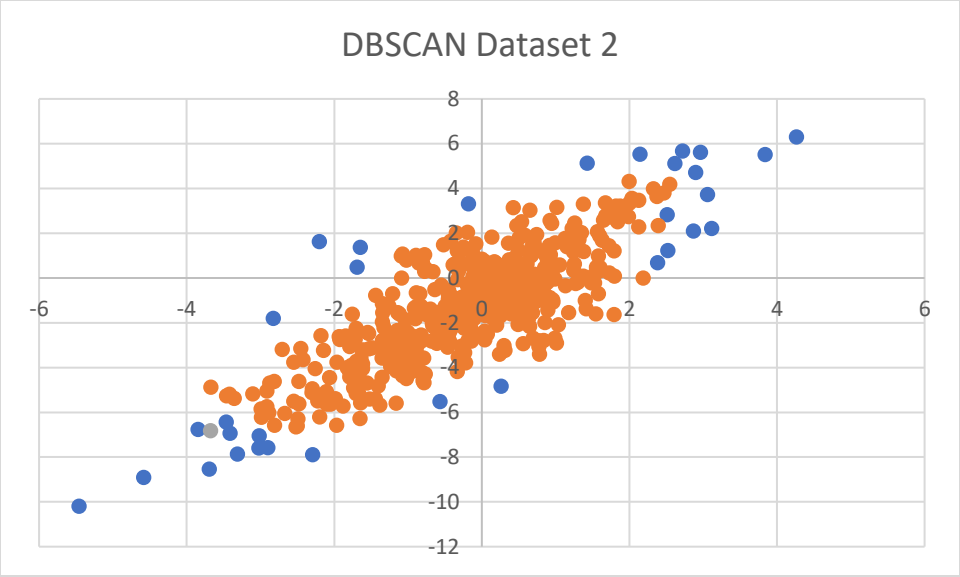
Why KMeans?

KMeans is simple, efficient, and is flexible and the time complexity is pretty good. The problem is that KMeans cannot deal with non-spherical clusters very well, clusters that do not have uniform size or density clusters, and you have to specify the number of clusters before the algorithm starts.



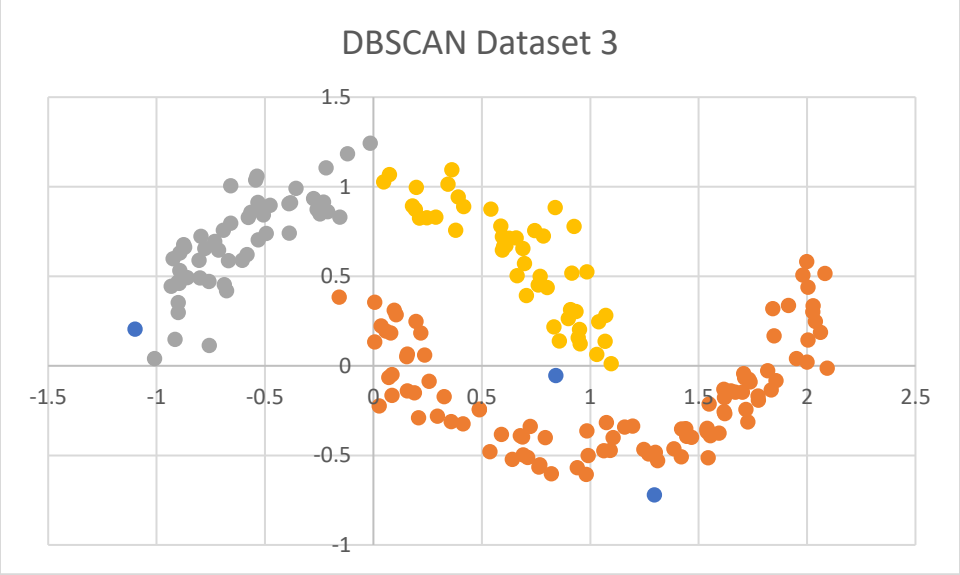

Purity: 0.926666

NMI: 0.9745




Purity: 0.708

NMI: 0.00525345

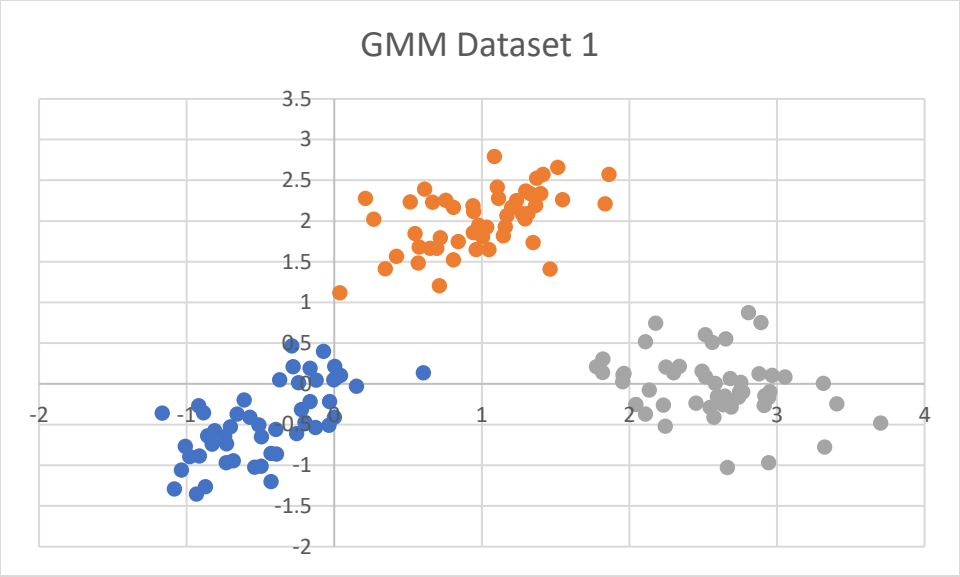



Purity: 0.985

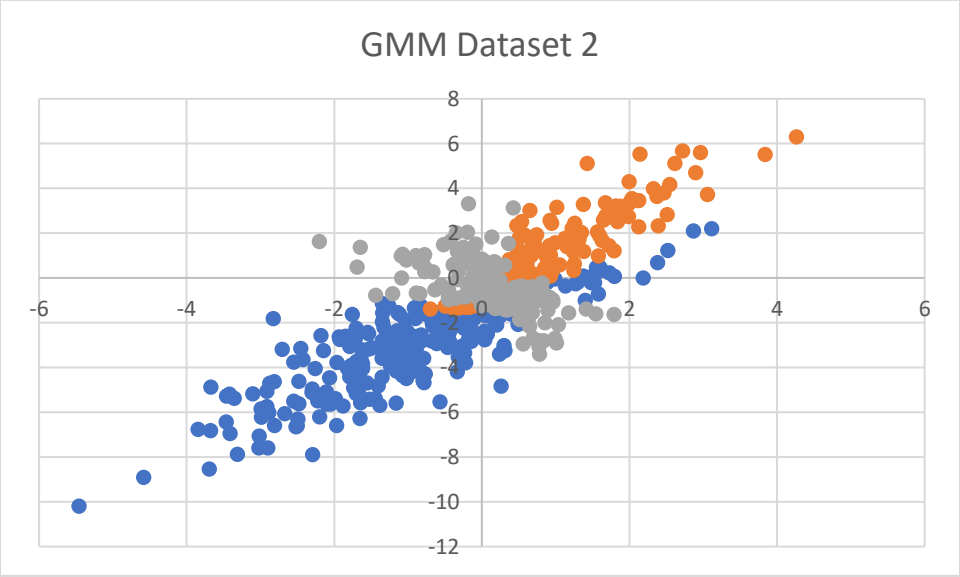
NMI: 0.81735

Why DBSCAN?

It can do clusters of arbitrary shapes and it is robust to noise. The problem is that it isn't very good if there aren't enough connected regions of sufficiently high density.

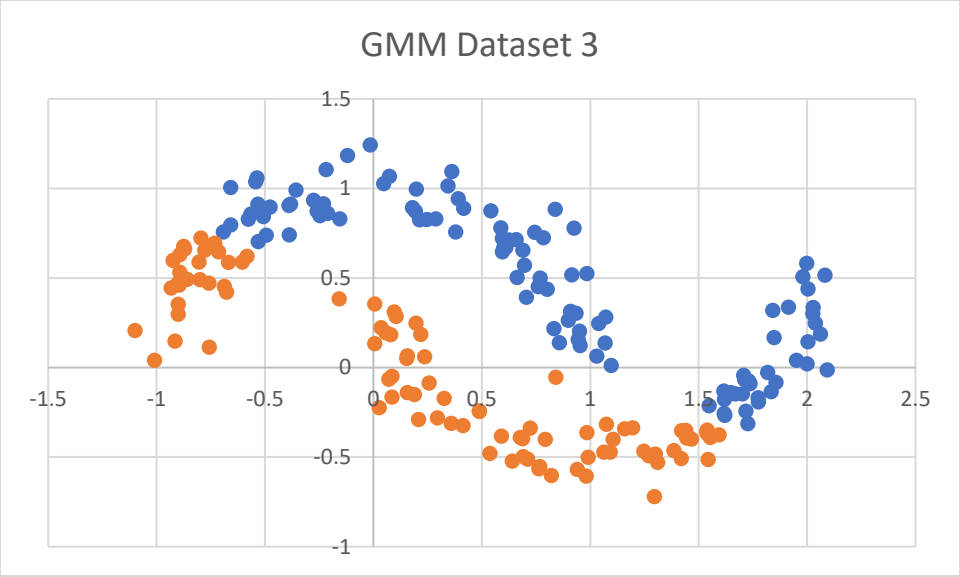



Purity: 1.0
NMI: 1.0




Purity: 0.764

NMI: 0.075603




Purity: 0.69

NMI: .0759484

Why GMM?

Probably has the best accuracy and is better at clusters of differing sizes and shapes. The problem of GMM is that the computation time is much longer and intensive compared to the other clustering methods.