Nick Holleran
Day Lecture
ID: 5222260

Project 3

Problem 1:

1)

a) Entropy = 0.55341029617178

```
Confusion Matrix: 0   1     2   3
                  5  137   1   0 | site1
                  2    7  43  60 | site2
                 25    0   0    0 | site3
                  48   3   0    1 | site4
```

According to the confusion matrix and entropy, site 1 is relatively separate from other classes and easy to identify, while sites 3 and 4 mostly belong to the same cluster, and thus are hard to identify between the two of them. Site 2 is split between multiple clusters and showing that perhaps 4 clusters is too many. The entropy for this clustering is quite low despite the confusion matrix, which shows that k means is consistent in putting each class in a cluster.

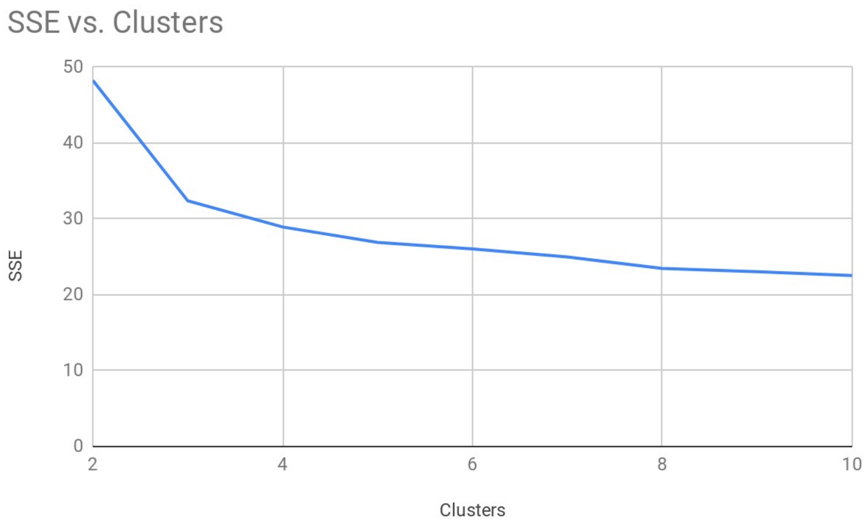
b) Entropy = 0.58476415225191

```
Confusion Matrix =
0   1     2  <-- assigned to cluster
5 137   1 | site1
2   9 101 | site2
25   0   0 | site3
48   3   1 | site4
```

Site 1 stays the same, as it didn't have any clusters in cluster 3, but site 2 which was split between clusters 2 and 3 is now assigned mostly to its own cluster, with only 11 points mis-assigned. Site 3 and site 4 are still assigned to the same cluster. Based on these results I can say the site 1 is the most easily identifiable, with site 2 also being identifiable, but maybe not as easy for k means to cluster. Site 3 and 4, however are very similar clusters and very hard to determine which one is which.

c)

Clusters	SSE
2	48.2581284681525
3	32.39322662912998
4	28.934177759717986
5	26.896807189992245
6	26.042648927820206
7	24.99144086954106
8	23.469818581309976
9	23.032877901145767
10	22.54485183968312



The plot above shows that the more clusters there are, the lower the SSE will be. This makes sense because the SSE is the sum of squared errors, or the location of each point

from the centroid, and if there are more centroids, then there is bound to be one centroid closer to the point, lowering the SSE.
2)
3 Clusters:
Confusion matrix: Entropy = 1.0238827013299

```
   0   1   2  <-- assigned to cluster
 143   0   0 | site1
  13  97   2 | site2
  19   0   6 | site3
  51   1   0 | site4
```

4 Clusters:
Confusion Matrix: Entropy = 0.99839783857657

```
   0   1   2   3  <-- assigned to cluster
 143   0   0   0 | site1
  11  97   2   2 | site2
  19   0   6   0 | site3
  51   1   0   0 | site4
```

a) Both methods (4 and 3 clusters) struggle to separate the classes. In both methods site 1, site 3, and site 4 all belong in the same cluster for the most part, site 2 is the only class not put in the original cluster. The 3rd (and 4th if used) clusters are almost completely empty. The entropies for both methods are also both extremely high when compared with the k-means algorithm.

b) K-means performs better than the hierarchical method is both cases, with the 3 cluster case being better for k-means. This is likely because k-means does well when trying to find clusters that are not well separated, but the hierarchical method is not able to find when one cluster stops and another starts.

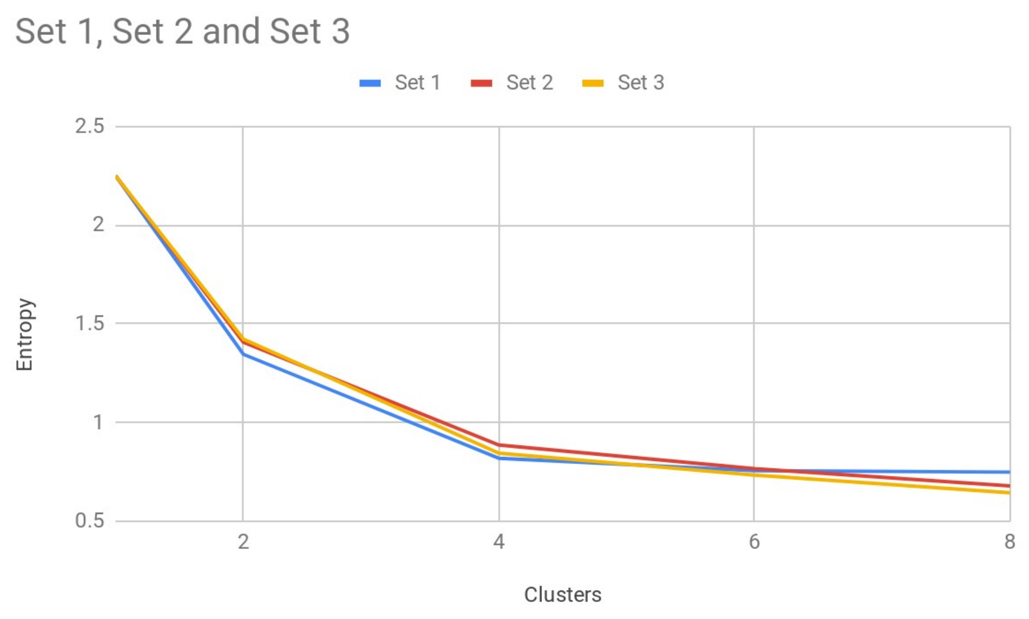
Problem 2:

a)

	Data Set 1	Data Set 2	Data Set 3
1 Cluster	0 50 | 1 100 | 2 100 | 3 50 | 4 100 | 5	0 50 | 1 100 | 2 100 | 3 50 | 4 100 | 5	0 50 | 1 100 | 2 100 | 3 50 | 4 100 | 5
2 Clusters	0 1 49 1 | 1 100 0 | 2 100 0 | 3 2 48 | 4 0 100 | 5	0 1 46 4 | 1 100 0 | 2 100 0 | 3 5 45 | 4 0 100 | 5	0 1 45 5 | 1 100 0 | 2 100 0 | 3 6 44 | 4 0 100 | 5
4 Clusters	0 1 2 3 22 27 1 0 | 1 100 0 0 0 | 2 0 100 0 0 | 3 0 2 43 5 | 4	0 1 2 3 28 18 4 0 | 1 0 100 0 0 | 2 100 0 0 0 | 3 1 4 42 3 | 4	0 1 2 3 22 23 4 1 | 1 100 0 0 0 | 2 0 100 0 0 | 3 1 5 29 15 | 4

	0 0 100 0 | 5	0 0 100 0 | 5	0 0 100 0 | 5
6 Clusters	0 1 2 3 4 5 22 27 1 0 0 0 | 1 100 0 0 0 0 0 | 2 0 100 0 0 0 0 | 3 0 2 30 6 5 7 | 4 0 0 100 0 0 0 | 5	0 1 2 3 4 5 28 18 2 2 0 0 | 1 0 100 0 0 0 0 | 2 100 0 0 0 0 0 | 3 1 4 10 21 11 3 | 4 0 0 0 100 0 0 | 5	0 1 2 3 4 5 22 23 4 1 0 0 | 1 100 0 0 0 0 0 | 2 0 100 0 0 0 0 | 3 1 5 4 15 18 7 | 4 0 0 0 0 100 0 | 5
8 Clusters	0 1 2 3 4 5 6 7 22 5 22 1 0 0 0 0 | 1 100 0 0 0 0 0 0 0 | 2 0 12 88 0 0 0 0 0 | 3 0 1 1 17 6 13 5 7 | 4 0 0 0 67 0 33 0 0 | 5	0 1 2 3 4 5 6 7 28 17 2 1 2 0 0 0 | 1 0 100 0 0 0 0 0 0 | 2 100 0 0 0 0 0 0 0 | 3 1 0 10 4 17 11 3 4 | 4 0 0 0 0 100 0 0 0 | 5	0 1 2 3 4 5 6 7 22 6 17 4 1 0 0 0 | 1 100 0 0 0 0 0 0 0 | 2 0 0 100 0 0 0 0 0 | 3 1 5 0 4 8 7 18 7 | 4 0 0 0 0 0 0 100 0 | 5

Clusters	Set 1	Set 2	Set 3
1	2.25	2.25	2.25
2	1.3453942556546	1.4063245121755	1.4222222215611
4	0.81872942946731	0.88603470049014	0.84470280569843
6	0.75630912841143	0.76666928423608	0.73375809253787
8	0.74881374116637	0.67859420039045	0.64391905660564



The above graph shows that entropy does indeed decrease as the number of clusters increases, this is true across all three data sets. This shows that the more clusters there are, the more likely each cluster will end up in a corresponding class, which makes sense given that there are 5 classes and 2 of them are noise. Set 1 has a more significant dip in entropy as we start to increase clusters but quickly loses ground to data sets 2 and 3, of which data set 3 ends up performing better. This is because the clusters in set 1 are less dense, so when calculating entropy there is a good probability it will end up in noise and the entropy won’t increase that much since noise and the clusters are related, but as we increase the amount of clusters, the real classes become more clear and the less dense graphs entropys decrease at a slower rate and the dense graphs entropys start dropping faster, this is seen in set 3 which is very dense, and set 1 which is more sparse.

b)
The noise tends to mess up with the average type of clustering, if the switch to MAX or complete link was made we would see the clustering algorithm work better against the noise in all the data sets.

Problem 3:
a)
Data Set 1:
Confusion Matrix: Entropy: 0.5677608757953

```
  0  1  2  3  4  5  6  <-- assigned to cluster
  5 12 12  2  1  0  0 | 1
 10  0 45 32  7  6  0 | 2
  0 99  0  0  0  0  0 | 3
  0  0  0  0  0  0 17 | 4
  0  0  0  0  0  0 96 | 5
```

Data Set 2: Entropy: 0.40197539311008

```
   0   1   2   3  <-- assigned to cluster
   9  11   0   0 | 1
 100   0   0   0 | 2
   0 100   0   0 | 3
   0   0   7   6 | 4
   0   0 100   0 | 5
```

Data Set 3: Entropy: 0.073582530310491

```
   0   1   2  <-- assigned to cluster
   0   2   0 | 1
 100   0   0 | 2
   0 100   0 | 3
   0   0   1 | 4
   0   0 100 | 5
```

Set 1 has the highest entropy of .56, followed by a slightly lower entropy in set 2 with .4, but set 3 has a very low entropy with .07. This corresponds well with the data given that set 1 and set 2 are less dense and so it is harder for the algorithm to differentiate between the clusters and the noise, but as the classes become more dense (the three classes that aren’t noise anyways) then the algorithm has a much easier time finding where each cluster stops and where noise begins, this helps with the epsilon being so small so that only very dense clusters are found.

b)
Data Set 1: Entropy: 2.1816873722255

```
   0   1  <-- assigned to cluster
  50   0 | 1
 100   0 | 2
 100   0 | 3
  36   6 | 4
 100   0 | 5
```

Data Set 2: Entropy: 0.76200402608886

```
   0   1   2   3   4  <-- assigned to cluster
  19  23   2   2   0 | 1
```

```
 100   0   0   0   0 | 2
   0 100   0   0   0 | 3
   0   0  38   2   4 | 4
   0   0 100   0   0 | 5
```

Data Set 3: Entropy: 0.4484185638514

```
   0   1   2   3   4   5   6   7  <-- assigned to cluster
  14  11   4   8   5   0   0   1 | 1
   0 100   0   0   0   0   0   0 | 2
   0   0   0 100   0   0   0   0 | 3
   6   0   1   0   0  10   6   5 | 4
   0   0   0   0   0 100   0   0 | 5
```

We see a similar comparison with each entropy as in the last part, with set 1 having the highest entropy followed by set 2 and then finally set 3. The difference this time is that the entropys are all much higher than they were in part a. This is because of the increase in epsilon. If we increase the radius of the circle in dbscan, it will pick up much more noise than before at the edge of each cluster, and each cluster might pick up the points that belong in other classes with an epsilon so high.

c)
The overall entropy for hierarchical clustering compared to DBSCAN is much higher, this shows that hierarchical clustering with AVERAGE link struggles heavily with noise and with varying densities (across varying data sets), while DBSCAN can handles both of these issues rather well. This is assuming that the correct parameters are selected for DBSCAN. The more dense a graph is, the better both DBSCAN and hierarchical clustering performs, this makes sense because each cluster is very clear and identifiable from each other and from the surrounding noise. Varying densities across the same data set was never explored, but this is something that both hierarchical struggle with so the entropys for both would be high. In conclusion the hierarchical clustering method does not handle the noise and outliers well while the DBSCAN algorithm does.