Homework-5

1(a) Given

$$l(\theta)=\sum_{i=1}^{n}\log\left[\sum_{j=1}^{k}P(x^{(i)}|\mu_j C_j)P(j)\right]$$

$$\frac{\partial}{\partial \mu_j}l(\theta)=\sum_{i=1}^{n}\frac{P(j)N(x^{(i)}|\mu_j,C_j)\nabla_{\mu_j}\{-\frac{1}{2}(x^{(i)}-\mu_j)^T C_j^{-1}(x^{(i)}-\mu_j)\}}{\sum_{j'=1}^{K}P(j')N(x^{(i)}|\mu_{j'},C_{j'})}$$

$$=\sum_{i=1}^{n}P_{i,j}C_j^{-1}(x^{(i)}-\mu_j)$$

1(b) $$\frac{\partial}{\partial P(j)}l(\theta)=\sum_{i=1}^{n}\frac{N(x^{(i)}|\mu_j C_j)}{\sum_{j'=1}^{K}P(j')N(x^{(i)}|\mu_{j'},C_{j'})}$$

$$=\frac{1}{P(j)}\sum_{i=1}^{n}P_{i,j}\ [\text{Given: } P_{i,j}=P(j|x^{(i)})]$$

No, it will not be a valid probability distribution

1(c) $\frac{\partial}{\partial P(j)} l(\theta) = \frac{1}{P(j)} \sum_{i=1}^{n} P_{i,j}$

$$P(j) = \frac{exp(w_j)}{\sum_{j'=1}^{K} exp(w_{j'})}$$

using lagrange multipliers,

$$P(j) = \frac{\sum_{i=1}^{n} P_{i,j}}{\lambda}$$

Summing over j and normalizing,

$$P(j) = \frac{\sum_{i=1}^{n} P_{i,j}}{N}$$

$$\Rightarrow \frac{\sum_{i=1}^{n} P_{i,j}}{P(j)} = N$$

$$\therefore \frac{\partial}{\partial w_j} l(\theta) \propto \sum_{i=1}^{n} P_{ij} - P(j)$$

Problem 1(d):

$$\frac{\partial}{\partial C_j} l(\theta) = \sum_{i=1}^{n} \frac{P(j) \nabla_{C_j} N(x^{(i)} | \mu_j, C_j)}{\sum_{j'=1}^{K} P(j') N(x^{(i)} | \mu_{j'}, C_{j'})}$$

Here $\nabla_{C_j} N(x | \mu_j, C_j) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|C_j|^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2}(x-\mu_j)^T C_j^{-1} (x-\mu_j) \right\}$

$$\nabla_{C_j} \cdot \left\{ \nabla_{C_j} \left(-\frac{1}{2}(x-\mu_j) C_j^{-1} (x-\mu_j) \right) - \right.$$

$$C_j^{-1} \nabla_{C_j} |C_j|]$$

$$= N(x | \mu_j, C_j) \nabla (\log N(x | \mu_j, C_j))$$

So, $$C_j = \frac{\sum_{i=1}^{n} P_{i,j} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{n} P_{i,j}}$$

No, the result won't stay positive.

1(e)

Advantages:

1) In EM-algorithm the proposed parameter values are always valid for example, probability masses between [0,1] sums to 1, which is not in the cases of gradient descent.

2) In EM-algorithm we don't have to calculate the likelihood to insure it has increased at every step which is not in the case while gradient descent.

3) EM method exploits structure of the objective and the variable involved in a manner that they are largely decoupled which allows good convergence rate than gradient descent.

Problem-2(a)(I)

Homogeneity Score: 0.419805

Completeness Score: 0.441756

V Measure Score: 0.430501

Adjusted Mutual Info Score- 0.430354

Adjusted Rand Score- 0.320217

Problem-2(a)(II)

Smallest Objective Value among 10: 15007778.664257059

Homogeneity Score: 0.422339

Completeness Score: 0.444161

V Measure Score: 0.432975

Adjusted Mutual Info Score- 0.432829

Adjusted Rand Score- 0.322483

Problem-2(b)(I)

10! Different matchings are possible between the clusters and the classes of data

Computational complexity of Hungarian algorithm: O(n^3)

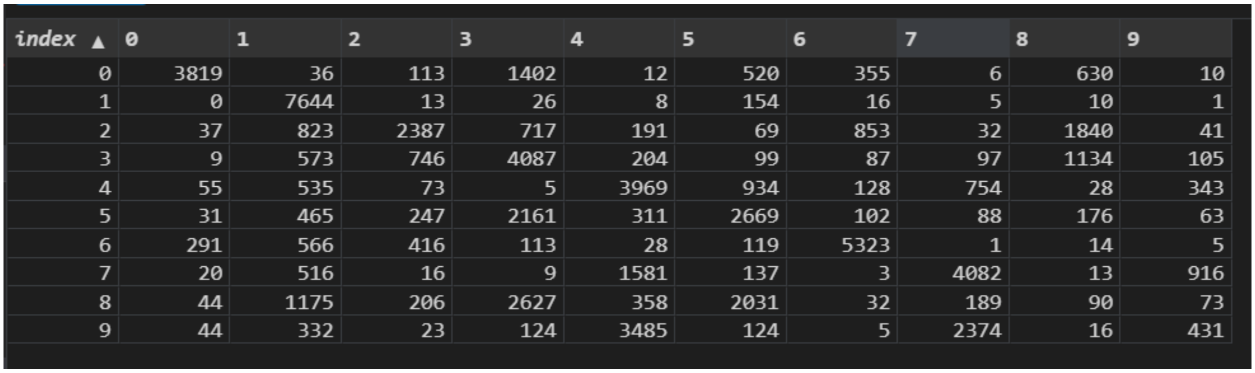
Problem-2(b)(II)

2.1.1

Mapping

{0: 0, 1: 9, 2: 2, 3: 6, 4: 1, 5: 3, 6: 8, 7: 7, 8: 5, 9: 4}

Confusion Matrix



index ▲	0	1	2	3	4	5	6	7	8	9
0	3819	36	113	1402	12	520	355	6	630	10
1	0	7644	13	26	8	154	16	5	10	1
2	37	823	2387	717	191	69	853	32	1840	41
3	9	573	746	4087	204	99	87	97	1134	105
4	55	535	73	5	3969	934	128	754	28	343
5	31	465	247	2161	311	2669	102	88	176	63
6	291	566	416	113	28	119	5323	1	14	5
7	20	516	16	9	1581	137	3	4082	13	916
8	44	1175	206	2627	358	2031	32	189	90	73
9	44	332	23	124	3485	124	5	2374	16	431

Accuracy

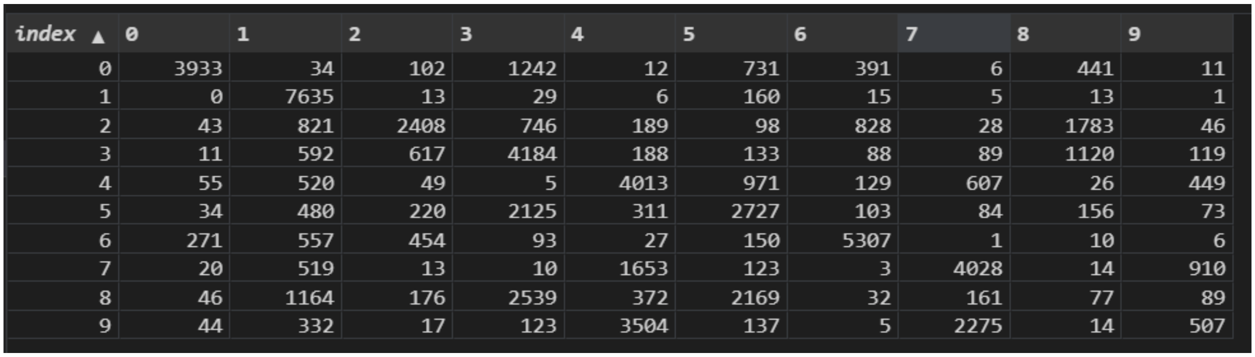
0.49287142857142857

2.1.2

Mapping

{0: 8, 1: 3, 2: 4, 3: 6, 4: 5, 5: 7, 6: 2, 7: 9, 8: 1, 9: 0}

Confusion Matrix



index ▲	0	1	2	3	4	5	6	7	8	9
0	3933	34	102	1242	12	731	391	6	441	11
1	0	7635	13	29	6	160	15	5	13	1
2	43	821	2408	746	189	98	828	28	1783	46
3	11	592	617	4184	188	133	88	89	1120	119
4	55	520	49	5	4013	971	129	607	26	449
5	34	480	220	2125	311	2727	103	84	156	73
6	271	557	454	93	27	150	5307	1	10	6
7	20	519	13	10	1653	123	3	4028	14	910
8	46	1164	176	2539	372	2169	32	161	77	89
9	44	332	17	123	3504	137	5	2275	14	507

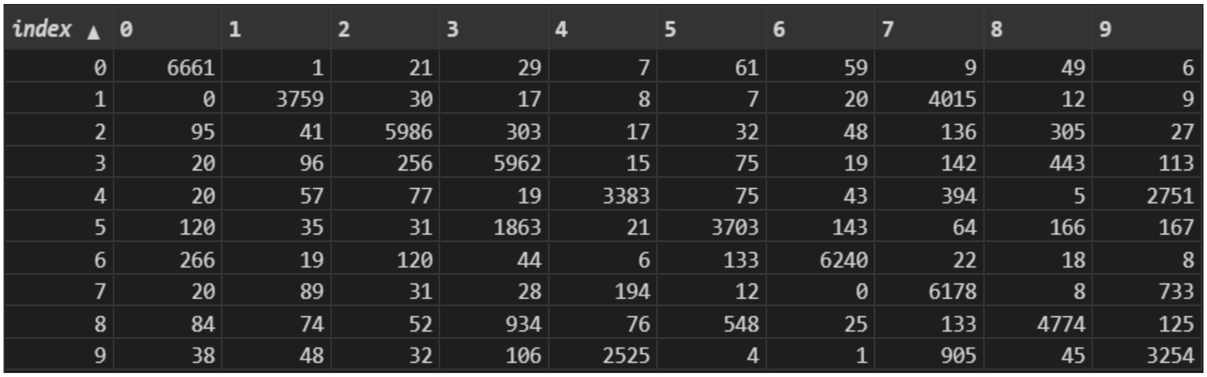
Accuracy

0.5023714285714286

Problem-2(c)

(i)

1. Achieved Assignment: {0: 5, 1: 2, 2: 4, 3: 7, 4: 3, 5: 8, 6: 6, 7: 1, 8: 0, 9: 9}
2. Confusion Matrix:



index ▲	0	1	2	3	4	5	6	7	8	9
0	6661	1	21	29	7	61	59	9	49	6
1	0	3759	30	17	8	7	20	4015	12	9
2	95	41	5986	303	17	32	48	136	305	27
3	20	96	256	5962	15	75	19	142	443	113
4	20	57	77	19	3383	75	43	394	5	2751
5	120	35	31	1863	21	3703	143	64	166	167
6	266	19	120	44	6	133	6240	22	18	8
7	20	89	31	28	194	12	0	6178	8	733
8	84	74	52	934	76	548	25	133	4774	125
9	38	48	32	106	2525	4	1	905	45	3254

3. Accuracy: 0.7128571428571429 (Accuracy changes based on the randomness of Sklearn Kmeans but it is always in 0.6-0.7 range)

(ii) Yes, Spectral clustering produces better accuracy. K-means clustering assumes the points assigned to a cluster are spherical about the cluster center. Spectral clustering helps create more accurate clusters when this assumption may not always be true by minimizing squared errors in the input domain on the ability to reconstruct neighbors.

Problem 2(d) (I):

Accuracy on K-means method:

For k = 1, 0.818412017167382

For k = 3, 0.8084835479256081

For k= 5, 0.7937195994277539

Accuracy on random sampling:

For k = 1, 0.6566523605150214

For k = 3, 0.6463090128755364

For k= 5, 0.6225178826895565

Accuracy on spectral clustering:

For k = 1, 0.7935479256080115

For k = 3, 0.7858798283261803

For k= 5, 0.7803576537911302

Problem 2(d) (II):

Random sampling + KNN doesn't give good results because the number of test set is greater than train set. K-means +KNN performs well because we choose the best 100 centroids as train set by implementing k-means algorithm first. Spectral clustering + KNN also performs well because in this case we first implement spectral clustering +k-means to choose test set and train set efficiently. Spectral clustering is a very strong clustering method where points are projected into a space of infinite dimensions and it still works when the clusters are not linearly separable.