**PROBLEM 1 MARKOV DECISION PROCESS**

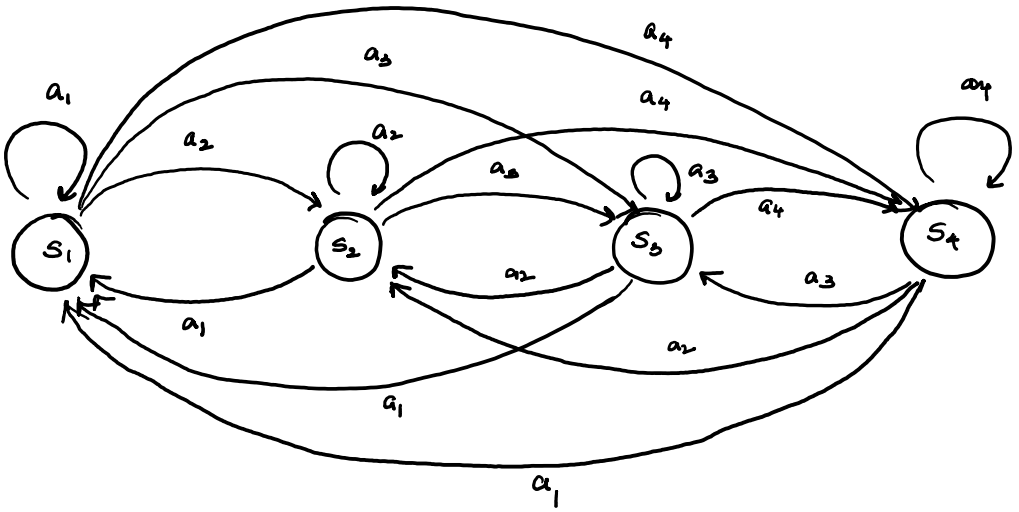
① Markov Decision Process (MDP)

↳ 4 states $s_1, s_2, s_3, s_4$

↳ 4 actions $a_1, a_2, a_3, a_4$

TRANSITIONS :

$T(s_i, a_j) = s_j$ for all $j \in \{1, 2, 3, 4\}$




REWARD FUNCTIONS :

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ |
|---|---|---|---|---|
| $s_1$ | 0 | 1 | 0.5 | 0.5 |
| $s_2$ | 0.5 | 0 | 1 | 0.5 |
| $s_3$ | -1 | 0.5 | 0 | 0.5 |
| $s_4$ | -1 | 0.5 | 0.5 | 0 |

1 a) Possible deterministic policies :

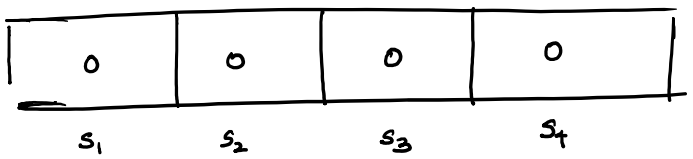
As we have 4 states & 4 actions

$\pi$ (Policy) is a mapping from states to actions.

There can be $\underline{\underline{4^4}}$ policies.

b) For $\gamma = 0.8$, Optimal value function $V^*$ and optimal policy $\pi^*$

INITIAL VALUE : $V(s) = 0 \quad \forall s \in S$




Using the greedy strategy

$$V_{t+1}(s_1) = \max_a \left( R(s_1, a) + \gamma V_t (T(s_1, a)) \right)$$ Possible actions $\{a_1, a_2, a_3, a_4\}$

$$\max_a \Big( R(s_1,a_1) + \gamma V_0 (T(s_1,a_1)),\ R(s_1,a_2) + \gamma V_0 (T(s_1,a_2)),\ R(s_1,a_3) + \gamma V_0 (T(s_1,a_3)),\ R(s_1,a_4) + \gamma V_0 (T(s_1,a_4)) \Big)$$ $\{s_1, s_2, s_3, s_4\}$

$$= \max_a \left( 0 + \gamma V_t(s_1),\ 1 + \gamma V_t(s_2),\ 0.5 + \gamma V_t(s_3),\ 0.5 + \gamma V_t(s_4) \right)$$

$$= \max_a \left( 0, 1, 0.5, 0.5 \right) = \boxed{\underline{1}}$$ For action $\boxed{a_2}$ $\boxed{\pi^1(s_1) = a_2}$

Similarly for $s_2$, $s_3$, $s_4$ we get

$$V_{t+1}(s_2) = \max\Big(R(s_2,a_1) + \gamma V_t(T(s_2,a_1)),\ R(s_2,a_2) + \gamma V_t(T(s_2,a_2))$$
$$R(s_2,a_3) + \gamma V_t(T(s_2,a_3)),\ R(s_2,a_4) + \gamma V_t(T(s_2,a_4))\Big)$$

$$= \max\Big(0.5 + \gamma V_t(s_1),\ 0 + \gamma V_t(s_2),\ 1 + \gamma V_t(s_3),\ 0.5 + \gamma V_t(s_4)\Big) \rightarrow \text{(I)}$$

$$V_{t+1}(s_3) = \max\Big(R(s_3,a_1) + \gamma V_t(T(s_3,a_1)),\ R(s_3,a_2) + \gamma V_t(T(s_3,a_2))$$
$$R(s_3,a_3) + \gamma V_t(T(s_3,a_3)),\ R(s_3,a_4) + \gamma V_t(T(s_3,a_4))\Big)$$

$$= \max\Big(-1 + \gamma V_t(s_1),\ 0.5 + \gamma V_t(s_2),\ 0 + \gamma V_t(s_3),\ 0.5 + \gamma V_t(s_4)\Big) \rightarrow \text{(II)}$$

$$V_{t+1}(s_4) = \max\Big(R(s_4,a_1) + \gamma V_t(T(s_4,a_1)),\ R(s_4,a_2) + \gamma V_t(T(s_4,a_2))$$
$$R(s_4,a_3) + \gamma V_t(T(s_4,a_3)),\ R(s_4,a_4) + \gamma V_t(T(s_4,a_4))\Big)$$

$$= \max\Big(-1 + \gamma V_t(s_1),\ 0.5 + \gamma V_t(s_2),\ 0.5 + \gamma V_t(s_3),\ 0 + \gamma V_t(s_4)\Big) \rightarrow \text{(III)}$$

$$V_{t+1}(s_1) = \max_a\Big(0 + \gamma V_t(s_1),\ 1 + \gamma V_t(s_2),\ 0.5 + \gamma V_t(s_3),\ 0.5 + \gamma V_t(s_4)\Big)$$
$$\rightarrow \text{(IV)}$$

Using equations (I), (II), (III), (IV) we can compute optimal policy as follows.

ITERATION - 1 : $\gamma = 0.8$ | $\gamma = 0.01$

$V_1(s_1) = \max(0, 1, 0.5, 0.5) = \boxed{1}$, $\pi_1(s_1) = a_2$ | $\boxed{1}$, $\pi_1(s_1) = a_2$

$V_1(s_2) = \max(0.5, 0, 1, 0.5) = \boxed{1}$, $\pi_1(s_2) = a_3$ (can be $a_3, a_4$) | $\boxed{1}$, $\pi_1(s_2) = a_3$

$V_1(s_3) = \max(-1, 0.5, 0, 0.5) = 0.5$, $\pi_1(s_3) = a_4$ | $0.5$, $\pi_1(s_3) = a_4$

$V_1(s_4) = \max(-1, 0.5, 0.5, 0) = 0.5$, $\pi_1(s_4) = a_4$ | $0.5$, $\pi_1(s_4) = a_4$

ITERATION → 2 VALUES @ 1 : 1 1 0.5 0.5

$\gamma = 0.8$

$V_2(s_1) = \max(\gamma, 1+\gamma, 0.5(\gamma+1), 0.5(\gamma+1)) = \max(0.8, 1.8, 0.9, 0.9) = 1.8 \ [a_2]$

$V_2(s_2) = \max(0.5+\gamma, \gamma, 1+0.5\gamma, 0.5(\gamma+1)) = \max(1.3, 0.8, 1.4, 0.9) = 1.4 \ [a_3]$

$V_2(s_3) = \max(-1+\gamma, 0.5+\gamma, 0.5\gamma, 0.5(\gamma+1)) = \max(-0.2, 1.3, 0.4, 0.9) = 1.3 \ [a_2]$

$V_2(s_4) = \max(-1+\gamma, 0.5+\gamma, 0.5(\gamma+1), 0.5\gamma) = \max(-0.2, 1.3, 0.9, 0.4) = 1.3 \ [a_2]$

From code we get optimal value function as follows:

$$\begin{bmatrix} 4.111111 & 3.8888 & 3.611111 & 3.611111 \end{bmatrix}$$

$s_1$ $s_2$ $s_3$ $s_4$

For policy $\pi^*$ we compute another round of iteration:

$V(s_1) = \max(0+4.111\gamma, 1+3.888\gamma, 0.5+3.6111\gamma, 0.5+3.61111\gamma)$ $\pi^*(s_1) = a_2$

$V(s_2) = \max(0.5+4.111\gamma, 3.888\gamma, 1+3.6111\gamma, 0.5+3.61111\gamma)$ $\pi^*(s_2) = a_3$

$V(s_3) = \max(-1+4.111\gamma, 0.5+3.888\gamma, 3.6111\gamma, 0.5+3.61111\gamma)$ $\pi^*(s_3) = a_2$

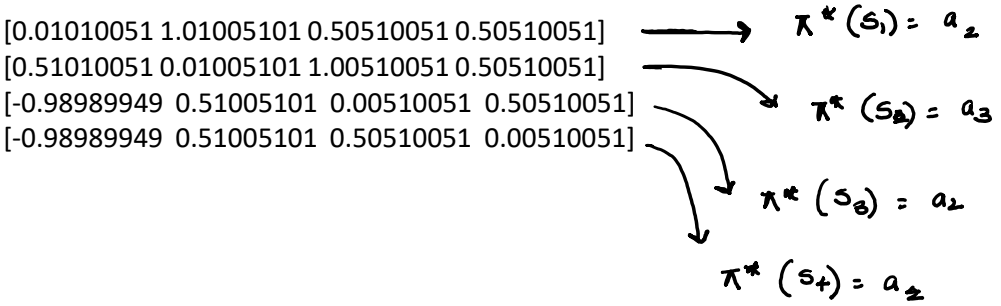
$V(s_4) = \max(-1+4.111\gamma, 0.5+3888\gamma, 0.5+3.614\gamma, 3.61111\gamma)$ $\pi^*(s_4) = a_2$

Yes there is a unique OPTIMAL DETERMINISTIC POLICY.

From code we get optimal value for $\gamma = 0.01$ as

$$\left[\ 1.01005101,\ 1.00510051,\ 0.510051,\ 0.510051\ \right]$$

Optimal policy can be found from.




Optimal policy did not change

2) For any MDP any two policies $\pi_1$ and $\pi_2$

Policy $\pi_3$ such that $V^{\pi_3}(s) \geq V^{\pi_1}(s)$

and $V^{\pi_3}(s) \geq V^{\pi_1}(s) \quad \forall s \in S$

This can be done using greedy strategy :

If we consider policy $\pi_3$ as

$$\pi(s) \in \arg\max_a \left( R(s,a) + \gamma V(T(s,a)) \right)$$

Then for any policy $\pi' \quad V^{\pi_3}(s) \geq V^{\pi'}(s) \quad \forall s \in S$

So, just consider $\pi_3$ as the optimal policy it automatically satisfies following constraints

$$V^{\pi_3}(s) \geq V^{\pi_1}(s)$$

$$V^{\pi_3}(s) \geq V^{\pi_2}(s)$$

3) Current Markov decision process requires

↳ Makes decision based on the current state of the environment and a choosen policy.

The current MDP :

STATE SPACE : A finite set of states (S)

ACTION SPACE : A set of allowable actions $A_s$ for each $s \in S$

TRANSITION FUNCTION : $T : S \times A \rightarrow S$

REWARD FUNCTION : $R : S \times A \rightarrow S$

Now if the MDP has to depend on last two states

STATE SPACE : A finite set of states (S) [REMAINS SAME]

ACTION SPACE : A set of allowable actions $A_s$ for each $s \in S$ [REMAINS SAME]

TRANSITION FUNCTION : $T : S'' \times S' \times A \rightarrow S$ [S', S'' represents previous two states]

REWARD FUNCTION : $R : S'' \times S' \times A \rightarrow S$

Now the decision depends on previous two states.

Yes this requirement can be formulated

**PROBLEM 2 GUASSIAN MIXTURES VS KMEANS**

1) Following are the scores obtained while minimizing kmeans objective function

| I\K | 12 | 18 | 24 | 36 | 42 |
|---|---|---|---|---|---|
| **1** | 268.677 | 198.703 | 480.839 | 177.143 | 107.203 |
| **2** | 204.742 | 193.9 | 137.76 | 262.072 | 90.719 |
| **3** | 275.314 | 266.191 | 480.839 | 135.508 | 92.2009 |
| **4** | 272.472 | 265.528 | 264.018 | 122.763 | 92.7352 |
| **5** | 484.256 | 139.611 | 192.355 | 262.058 | 81.8285 |
| **6** | 275.11 | 192.755 | 125.537 | 177.547 | 123.13 |
| **7** | 485.062 | 264.463 | 481.258 | 123.203 | 91.5404 |
| **8** | 202.348 | 270.771 | 481.189 | 262.072 | 122.625 |
| **9** | 214.529 | 264.681 | 264.239 | 122.822 | 97.352 |
| **10** | 275.11 | 140.005 | 481.258 | 189.945 | 88.8199 |
| **11** | 272.059 | 138.927 | 179.34 | 177.617 | 94.8996 |
| **12** | 202.808 | 193.445 | 481.067 | 262.072 | 65.4954 |
| **13** | 206.485 | 270.775 | 125.537 | 135.513 | 93.4978 |
| **14** | 277.777 | 199.351 | 264.091 | 135.513 | 88.6198 |
| **15** | 488.415 | 139.777 | 101.901 | 92.11 | 108.343 |
| **16** | 150.538 | 144.194 | 264.018 | 189.903 | 94.4921 |
| **17** | 485.269 | 198.648 | 198.681 | 135.476 | 93.974 |
| **18** | 484.214 | 265.528 | 198.817 | 189.928 | 104.102 |
| **19** | 272.059 | 266.191 | 118.628 | 478.855 | 95.2294 |
| **20** | 145.936 | 194.043 | 264.018 | 189.975 | 177.006 |

Means and variances for the objective function

| K | Mean | Variance |
|---|---|---|
| **12** | 297.159 | 13381 |
| **18** | 210.374 | 2572.19 |
| **24** | 279.269 | 20028.6 |
| **36** | 191.105 | 6981 |
| **42** | 100.191 | 466.285 |

2) Following are the log likelihood scores obtained while maximizing log likelihood using EM algorithm

| I\K | 12 | 18 | 24 | 36 | 42 |
|---|---|---|---|---|---|
| **1** | 13553.1 | 15082.7 | 15225.1 | 15098.8 | 15813 |
| **2** | 12581 | 14516.3 | 15560.6 | 15869.4 | 16006.6 |
| **3** | 13963 | 14411.8 | 14923.7 | 16507.2 | 17107.4 |
| **4** | 13610.3 | 14121.3 | 14486.7 | 16513.9 | 16660 |
| **5** | 13362.2 | 13772.2 | 15086.2 | 16150.3 | 16373.2 |
| **6** | 13131.5 | 13952.7 | 14955.8 | 15822.1 | 16067.1 |
| **7** | 11978.2 | 14719.9 | 14549.1 | 16095.1 | 16843 |
| **8** | 13878.4 | 14428.9 | 15373.1 | 15365.9 | 16455.1 |
| **9** | 13735.9 | 14836.7 | 14048.6 | 16093.1 | 16456.9 |
| **10** | 13680.7 | 14336.5 | 15042.2 | 15884.1 | 16362.8 |
| **11** | 13001.3 | 14296.5 | 14777.1 | 15973.1 | 15795 |
| **12** | 12364.2 | 14460.3 | 15008.2 | 16176.6 | 16333.2 |
| **13** | 13784.2 | 14835.2 | 14306.4 | 16210.7 | 15897.7 |
| **14** | 13067.5 | 14473.1 | 15488.5 | 15971.6 | 15945.7 |
| **15** | 13447.9 | 14334 | 15014.9 | 16443.3 | 16256.3 |
| **16** | 13812.2 | 14504.4 | 15367.6 | 15946 | 16583.5 |
| **17** | 12697.3 | 13604.5 | 15168.2 | 15328.9 | 16550.5 |
| **18** | 13544.3 | 14086.5 | 15188.6 | 15707.7 | 16390.7 |
| **19** | 13766.6 | 13418.5 | 14961.7 | 16080.1 | 16876.8 |
| **20** | 13755.6 | 13757.8 | 15476.5 | 15780.2 | 16523.8 |

Following are the mean and variances

| K | Mean | Variance |
|---|---|---|
| **12** | 13335.8 | 296295 |
| **18** | 14297.5 | 178522 |
| **24** | 15000.4 | 154516 |
| **36** | 15950.9 | 132848 |
| **42** | 16364.9 | 125347 |

3) Looking at the true labels

**TRUE LABELS**

[ 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 3. 3. 3. 3. 3. 3. 3. 3. 3. 3. 4. 4. 4. 4.
4. 4. 4. 4. 5. 5. 5. 5. 5. 5. 5. 5. 5. 5. 5. 5. 6. 6. 6. 6. 6. 6. 6. 6. 7. 7. 7. 7. 7. 7. 7. 7. 7. 7. 8. 8.
8. 8. 8. 8. 8. 8. 8. 8. 8. 9. 9. 9. 9. 9. 9. 9. 9. 9. 9. 9. 9. 9. 10. 10. 10. 10. 10. 10. 10. 10. 10. 10. 10.
10. 10. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 11. 12. 12. 12. 12. 12. 12. 12. 12. 12. 12. 12.
13. 13. 13. 13. 13. 13. 13. 13. 13. 13. 13. 13. 13. 14. 14. 14. 14. 14. 14. 14. 14. 14. 14. 14. 14. 15. 15. 15. 15.
15. 15. 15. 15. 15. 15. 22. 22. 22. 22. 22. 22. 22. 22. 22. 22. 22. 22. 23. 23. 23. 23. 23. 23. 23. 23. 23. 23. 24. 24.
24. 24. 24. 24. 24. 24. 24. 24. 24. 24. 25. 25. 25. 25. 25. 25. 25. 25. 25. 26. 26. 26. 26. 26. 26. 26. 26. 26. 26.
26. 26. 27. 27. 27. 27. 27. 27. 27. 27. 27. 27. 27. 28. 28. 28. 28. 28. 28. 28. 28. 28. 28. 28. 28. 29. 29. 29. 29.
29. 29. 29. 29. 29. 29. 29. 30. 30. 30. 30. 30. 30. 30. 30. 30. 30. 30. 30. 31. 31. 31. 31. 31. 31. 31. 31. 31. 31.
32. 32. 32. 32. 32. 32. 32. 32. 32. 32. 32. 33. 33. 33. 33. 33. 33. 33. 33. 33. 33. 33. 34. 34. 34. 34. 34. 34. 34.
34. 34. 34. 35. 35. 35. 35. 35. 35. 35. 35. 35. 35. 36. 36. 36. 36. 36. 36. 36. 36. 36. 36.]

**LABELS PREDICTED BY SCIKIT KMEANS FOR K = 36**

[ 1 1 1 1 1 1 1 1 1 1 1 24 24 24 1 24 24 1 24 24 24 16 1 1 1 1 1 1 1 1 16 1 1 1 1 1 1 16 1 1 1
24 24 24 24 24 24 1 16 16 9 9 9 9 9 9 9 9 1 1 24 1 1 1 24 1 1 1 33 34 29 34 34 31 30 33 36 32 28 16 1
1 16 1 1 16 1 16 16 1 1 1 1 16 16 16 16 16 16 16 16 16 16 16 16 16 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 24
24 24 24 24 24 24 24 24 24 24 24 24 1 1 1 1 16 16 1 16 1 1 16 16 16 24 24 24 24 1 24 24 24 24 24 24 24 16 16
16 16 16 16 16 16 16 16 24 24 24 24 9 24 1 24 1 1 9 24 1 1 9 1 9 9 1 1 1 1 1 1 1 1 1 1 16 1 1 1 1 1
1 1 16 16 16 16 16 16 16 16 16 16 1 1 1 1 1 1 16 1 24 1 1 1 1 1 1 1 1 1 24 1 1 1 1 24 24 24 24 24 24 24
24 24 24 24 24 24 1 1 1 1 1 1 1 1 1 1 1 1 16 16 16 1 16 16 16 1 16 16 16 1 26 8 5 22 6 10 21 23 25 20 27 1
1 1 24 24 24 24 1 1 1 1 1 16 1 1 16 24 16 1 16 16 16 15 18 7 3 4 14 11 13 12 17 2 24 24 24 24 24 24 24
24 24 24 24 24 16 9 16 16 16 16 16 9 9]

**LABELS PREDICTED BY GMM FOR K = 36**

Considering all the posterior probabilities for each data point corresponding to cluster.
Structure of clusters seem to be in conformance with the true labels

I found that data points attached to a cluster is more in the case of GMM than Kmeans.
That is Kmeans resulted in predicting lot less unique clusters than GMM.

Also considering the probabilistic nature of GMMs, I examined posteriors of the labels assigned to the cluster and found it to be better than Kmeans for my set of initializations

4) Following are results obtained while minimizing the objective function in Kmeans using Kmeans++ initialization technique

| I\K | 12 | 18 | 24 | 36 | 42 |
|---|---|---|---|---|---|
| **1** | 121.806 | 78.2529 | 61.0042 | 42.0613 | 27.5014 |
| **2** | 129.522 | 82.6997 | 61.8772 | 32.3626 | 27.3623 |
| **3** | 124.167 | 78.5555 | 59.0813 | 36.5231 | 31.0675 |
| **4** | 120.233 | 77.6419 | 61.6788 | 35.404 | 28.6775 |
| **5** | 126.35 | 78.1217 | 62.7954 | 40.5803 | 28.8933 |
| **6** | 124.404 | 76.574 | 63.416 | 34.4522 | 28.9894 |
| **7** | 124.827 | 86.8213 | 51.9504 | 33.3898 | 30.8838 |
| **8** | 129.166 | 80.6884 | 57.3095 | 34.4849 | 29.2948 |
| **9** | 120.055 | 85.5303 | 52.8248 | 36.6004 | 27.4801 |
| **10** | 127.999 | 79.4007 | 57.4575 | 35.558 | 28.3775 |
| **11** | 122.848 | 85.5132 | 59.3086 | 33.4525 | 29.9275 |
| **12** | 127.64 | 76.0005 | 63.9146 | 36.0513 | 27.9629 |
| **13** | 120.491 | 82.6734 | 66.9678 | 34.1654 | 29.2868 |
| **14** | 127.196 | 87.4601 | 64.1274 | 38.2736 | 29.3475 |
| **15** | 132.303 | 79.3483 | 54.4619 | 39.2613 | 27.8852 |
| **16** | 116.631 | 77.9335 | 60.5868 | 37.8807 | 29.3994 |
| **17** | 121.32 | 80.0351 | 58.5834 | 35.2679 | 31.099 |
| **18** | 120.432 | 72.7445 | 55.0605 | 34.769 | 26.8097 |
| **19** | 123.966 | 81.2104 | 54.3729 | 36.5796 | 29.1137 |
| **20** | 120.918 | 75.5943 | 59.9951 | 37.4158 | 37.7204 |

Following are the means and variances for Kmeans++

| K | Mean | Variance |
|---|---|---|
| **12** | 124.114 | 15.182 |
| **18** | 80.14 | 14.9107 |
| **24** | 59.3387 | 15.9693 |
| **36** | 36.2267 | 5.81012 |
| **42** | 29.354 | 5.09737 |

Following are the log likelihood scores for the Gaussian mixture model using Kmeans++ initialization

| I\K | 12 | 18 | 24 | 36 | 42 |
|---|---|---|---|---|---|
| **1** | 14094.5 | 14708.6 | 15528.1 | 16881.6 | 17486.5 |
| **2** | 14087 | 14848.3 | 15342 | 17097.7 | 17404.1 |
| **3** | 14145.6 | 14709 | 15520 | 17179.4 | 17479.9 |
| **4** | 14022.2 | 15081.3 | 15822.5 | 16926.5 | 17414.6 |
| **5** | 13910 | 14883.6 | 15711.1 | 16934.3 | 17445.4 |
| **6** | 13946.7 | 15042.1 | 15550.1 | 16850.9 | 17471.7 |
| **7** | 14063.2 | 14876.1 | 15545.2 | 16667.8 | 17562.4 |
| **8** | 14004.8 | 14818.3 | 15643.9 | 17191.6 | 17564.9 |
| **9** | 13921.8 | 14673.6 | 15897.4 | 16674.9 | 17527 |
| **10** | 13926.6 | 14905.6 | 15737.5 | 17186.3 | 17327.4 |
| **11** | 13965.4 | 14961.3 | 15340.1 | 16854.1 | 17635.2 |
| **12** | 14199.5 | 14829.7 | 15610.5 | 17043.8 | 17737 |
| **13** | 13753.8 | 14553.9 | 15747.8 | 16951.4 | 17354 |
| **14** | 14076.3 | 14959.6 | 15573.8 | 16740.7 | 17554.5 |
| **15** | 14166.6 | 14923.8 | 15502.7 | 16857.1 | 17523.4 |
| **16** | 14144.4 | 15103.1 | 15637.2 | 16964.8 | 17407.6 |
| **17** | 14080.2 | 14971.3 | 15721 | 16690.6 | 17669.1 |
| **18** | 14092.2 | 15075.1 | 15486.1 | 17073.8 | 17746 |
| **19** | 14191 | 15048.6 | 15610.3 | 16856.1 | 17405.2 |
| **20** | 14319 | 14980.6 | 15622.5 | 17094.3 | 17586.4 |

Following are the means and variances obtained using Kmeans++ initialization

| K | Mean | Variance |
|---|---|---|
| **12** | 14055.5 | 15611.7 |
| **18** | 14897.7 | 21285.3 |
| **24** | 15607.5 | 19143.7 |
| **36** | 16935.9 | 26907.8 |
| **42** | 17515.1 | 13411.8 |

Yes there is tremendous improvement. **Variance of the observations decreased a lot also in case of Kmeans inertia is minimized and in case of GMMs log likelihood is optimized.**

5) Covariance matrix of each mixture component is a diagonal matrix.

E-STEP of EM algorithm doesnot change i.e

$$q^{t+1} \in \underset{q_1, q_2, \ldots q_N}{\arg\max} \sum_{i=1}^{N} \sum_{y=1}^{K} q_i(y) \log \frac{P(x^{(i)}, Y=y|\theta)}{q_i(y)}$$

using lagrange multipliers for the constraint $\sum_y q_i(y) = 1$ gives

$$q_i^{(t+1)}(y) = p\left(Y=y \mid X=x^{(i)}, \theta^t\right)$$

M-STEP :

$$\theta^{t+1} \in \underset{\theta}{\arg\max} \sum_{i=1}^{N} \sum_{y=1}^{k} q_i^{t+1}(y) \log \frac{P(x^{(i)}, Y=y|\theta)}{q_i^{t+1}(y).}$$

Here $\theta$ involves $\mu, \Sigma, \lambda$

$\mu, \lambda$ also remains same

$$\rightarrow \sum_{i=1}^{N} \sum_{y=1}^{k} q_i^{t+1}(y) \log \left( \frac{\lambda_k \mathcal{N}(x^{(i)} \mid \mu_k, \Sigma_k')}{q_i^{(t+1)}(y)} \right)$$

Assuming each of $\Sigma_k'$ as a diagonal matrix i.e $\text{diag}(\sigma_{1k}'', \sigma_{2k}'', \ldots \sigma_{mk}'')$

We need to estimate each of these $\sigma_{jk}''$ for next Iteration

Inorder to estimate each of these $\sigma_{mk}^2$ parameters.

Lets follow the same procedure of maximizing the objective. We shall perform $\frac{\partial L}{\partial \sigma}$ and check for the optimal value.

$$\frac{\partial L}{\partial \sigma_{ab}} = \sum_{i=1}^{N} \left[ q_i^t(b) \frac{\frac{\partial}{\partial \sigma_{ab}} \mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b)}{\mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b)} \right]$$

Consider multivariate normal distribution

$$\mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b) = \frac{1}{\sqrt{(2\pi)^d \prod_{m=1}^{M} \sigma_{mb}}} \exp\left( -\frac{1}{2} (x^{(i)} - \mu_b) \Sigma_b^{-1} (x^{(i)} - \mu_b)^T \right)$$

$\Sigma_b = \text{diag}(\sigma_{1b}^2, \dots \sigma_{mb}^2)$ As determinant of diagonal matrix is product of diagonal elements.

$\Sigma_b^{-1} = \text{diag}\left( \frac{1}{\sigma_{1b}^2}, \dots \frac{1}{\sigma_{mb}^2} \right)$

$$\Rightarrow -\frac{1}{2} (x^{(i)} - \mu_b) \Sigma_b^{-1} (x^{(i)} - \mu_b)^T = \sum_{m=1}^{M} -\frac{1}{2\sigma_{mb}^2} (x_m^{(i)} - \mu_{mb})^2$$

Now, $\frac{\partial}{\partial \sigma_{ab}} \mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b) = \left[ \mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b) \frac{\partial}{\partial \sigma_{ab}} \left( \sum_{m=1}^{M} -\frac{1}{2\sigma_{mb}^2} (x_m^{(i)} - \mu_{mb})^2 \right) - \mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b) \frac{1}{\sigma_{ab}} \right]$

$$= \mathcal{N}(x^{(i)} \mid \mu_b, \Sigma_b) \left[ \frac{1}{\sigma_{ab}^3} (x_a^{(i)} - \mu_{ab})^2 - \frac{1}{\sigma_{ab}} \right] \longrightarrow (*)$$

Now,

$$\frac{\partial L}{\partial \sigma_{ab}} = \sum_{i=1}^{N} q_i^t(b) \left[ \frac{1}{\sigma_{ab}^3} \left( x_a^{(i)} - \mu_{ab} \right)^2 - \frac{1}{\sigma_{ab}} \right] = 0.$$

→ (+ve) [Most of the times, else sum shall remain +ve]

$$\sigma_{ab}^2 = \frac{\sum_{i=1}^{N} q_i^t(b) \left[ x_a^{(i)} - \mu_{ab} \right]^2}{\sum_{i=1}^{N} q_i^t(b)}$$

(+ve)

So, it is positive

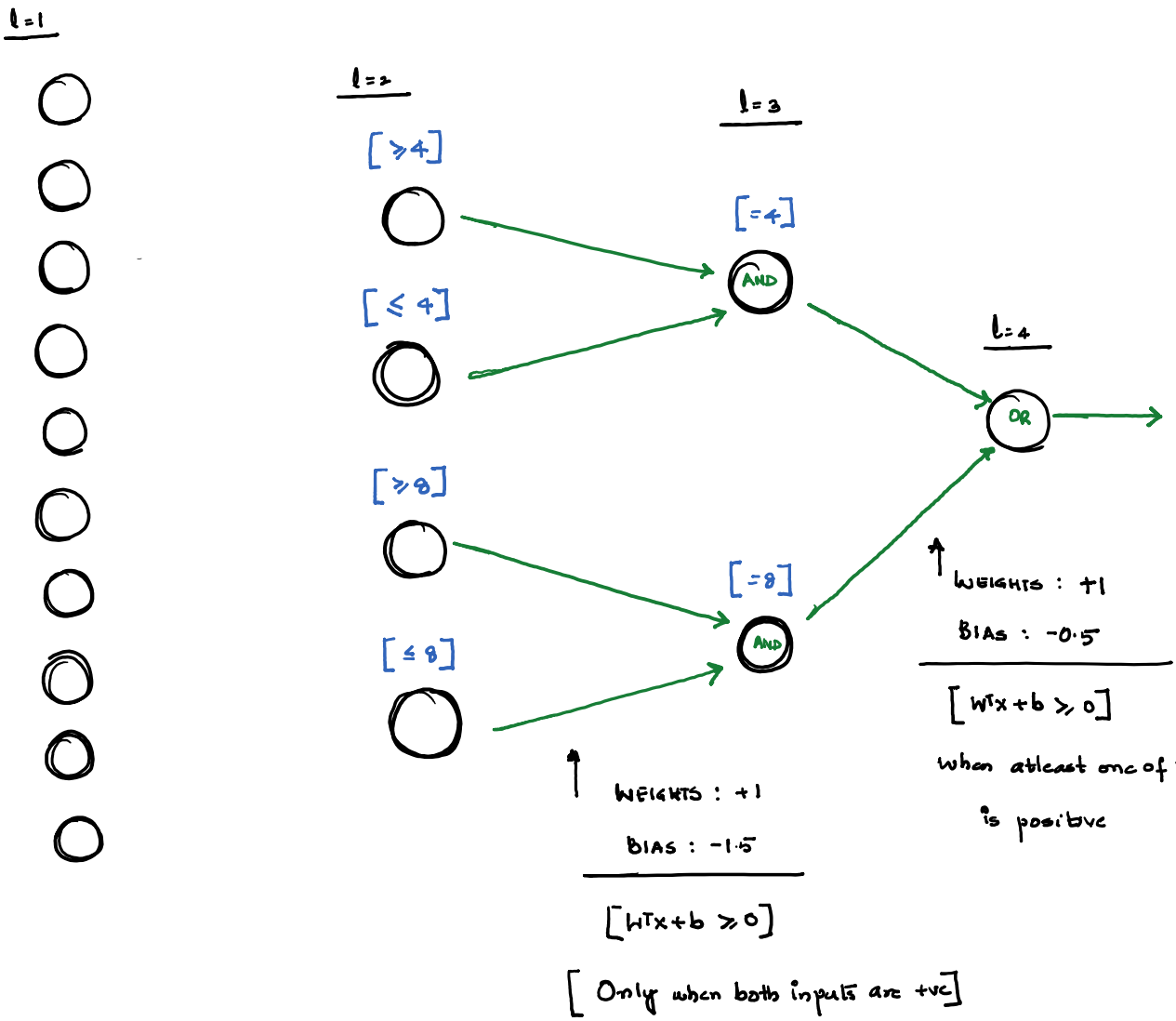
Thus in the EM procedure instead of considering the $\Sigma_k$ matrix as whole, we shall be considering individual $\sigma_{ab}^2$ and update these params.

i) No. of 1's in the input → divisible by (4) → 1 , 0 otherwise

**IDEA:** For equality case we can have two hidden units

(1) $w^T x + b \geq 0$ (2) $w^T x + b \leq 0$ (or) $-(w^T x + b) \geq 0$

**NETWORK STRUCTURE:**



Weight scheme for node $\geq 4$

WEIGHTS : +1.0

BIAS : −3.9

$[w^T x + b \geq 0]$

Only when atleast four of the inputs are +1's

Weight scheme for node $\leq 4$

WEIGHTS : −1.0

BIAS : +4.1

$w^T x + b \geq 0$

Only when atmost four of the inputs are +1's

Similarly weight scheme for node $\geq 8$

WEIGHTS : +1.0

BIAS : −7.9

$[w^T x + b \geq 0]$

Only when atleast 8 nodes are 1's

weight scheme for node $\leq 8$

WEIGHTS : −1.0

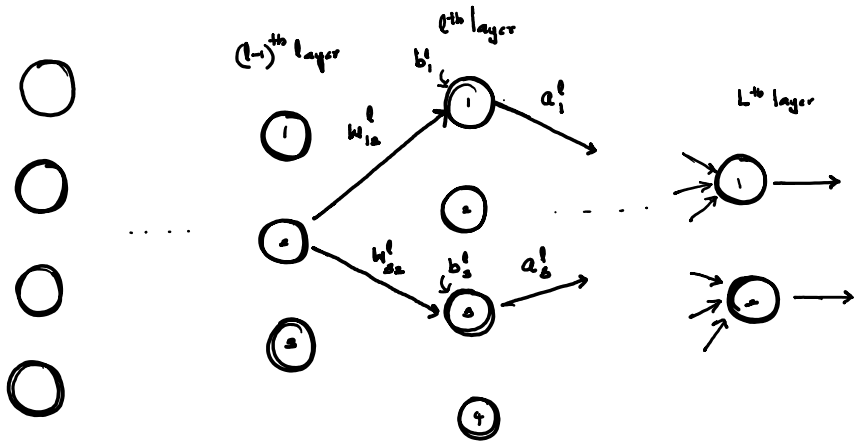
BIAS : +8.1

$[w^T x + b \geq 0]$

Only when atmost 8 nodes are 1's

(b) Adding an l2-norm $\lambda/2 \|w\|_2^2$ to the squared loss minimization

Considering same units as class. [Fitting all sigmoid units]



[Weighted input]

Activation can be given by $a_j^l = s\left(z_j\right)$ → [SOFT PLUS ACTIVATION]

$$= s\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right)$$

$$C(w,b) = \frac{1}{2}\left\| y - a(x,w,b)\right\|^2 + \lambda/2 \|w\|_2^2$$

$$\Rightarrow \quad C(w,b) = C_0(w,b) + \frac{\lambda}{2}\|w\|_2^2$$

[UNREGULARIZED COST FUNCTION]

$$\frac{\partial C}{\partial w} = \frac{\partial C_0}{\partial w} + \lambda w \quad , \quad \frac{\partial C}{\partial b} = \frac{\partial C_0}{\partial b}$$

⇒ For biases gradient descent rule doesnot change

$$b \to b - \eta \frac{\partial C_0}{\partial b}$$

Learning rule for weights becomes.

$$w \longrightarrow w - \eta \frac{\partial C_0}{\partial w} - \eta\lambda w$$

$$= w(1 - \eta\lambda) - \eta \frac{\partial C_0}{\partial w}$$

[WEIGHT DECAY]

So, on adding an L2-regularizer →

Update formula for bias doesnot change

But, for weights we need to consider weight decay

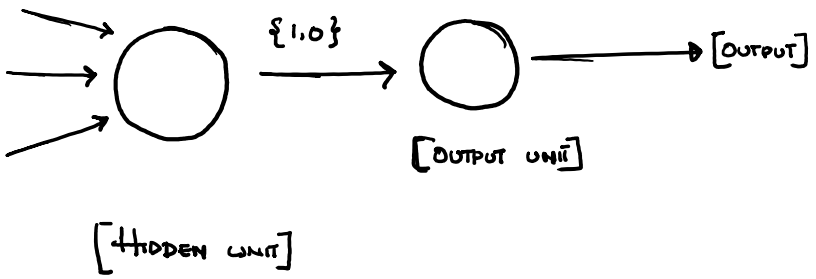
(3) Binary classification task :

Difference between a ① Single perceptron hidden unit

② Perceptron algorithm.

Perceptron algorithm : finds a linear classifier $[W^T x + b]$

But with a perceptron hidden unit




In perceptron algorithm :

It is trained by using perceptron loss. i.e

$$\nabla_w = -\sum_{m=1}^{M} y^{(m)} x^{(m)} \cdot 1_{-y^{(m)} f_{w,b}(x^{(m)}) \geq 0}$$

But for perceptron as hidden unit

Loss function → if out put is assumed as linear with weight [1]

[Misclassification loss]

So, weights learnt shall be different.

**BONUS**

After running the neural network on increasing size of the dataset.
The data showed to conform to the estimates of the perceptron algorithm.

I constructed a NN based on the same structure as considered for perceptron algorithm

I used **sigmoid units and used Adam optimizer.**
I calculated values with **two loss functions.**

1. **Cross entropy** which obligated me to create two output units for classes (0,1)
2. **BCELogitLoss** with which I found the following results

Following are the observations made after running on a datasets of size ranging from 1000 to 1000000

1. Output unit in the 4th layer as described in perceptron algorithm learned the following weights and biases to be an OR unit

   BIAS : -0.0859
   WEIGHTS : [0.1053, 0.1789]

2. Two units in the hidden layer 2 learned the following weights and biases [similar to an AND unit]

   BIAS : [-0.4021, -0.3673]
   WEIGHTS : [0.3045, 0.2978, 0.3458, 0.3010]

3. Four units in the hidden layer 1 learned the following weights and biases conforming to the strategy of [>=4, <=4, >=8, <=8] units

I also attached the code in file named BONUS.py