1.

a) The loss function now is $\sum_{i=1}^{m}(y^i - w^T x^i)^2$

b) The update rule is $w_i \leftarrow w_i + \alpha \sum_j x_{ij}(y^i - \sum_i w_i x_{ij})$

c) In fact, a neural network, which computes the output as, $y = round(g(w^T x))$

can be just treated as a processor of **Logistic regression**, which is **linear regression** essentially. So it can be solved analytically such as linear regression.

d)

Depending on the hint, the solution to calculate the vector $\mathbf{y}$ that minimizes

$$\|\mathbf{Ay} - \mathbf{b}\|^2$$

for a coefficient matrix $\mathbf{A}$ and a bias vector $\mathbf{b}$ is

$$\mathbf{y} = \mathbf{A}^+\mathbf{b}$$

In the current scene, the destination loss function is

$$\sum_{i=1}^{m}(y_i - g(w^T x^i))^2$$

To calculate the vector $\mathbf{y}$ that minimizes the loss function base on Eq.(1) and Eq.(3), we can know

$$\mathbf{A} = \mathbf{E}$$

$$\mathbf{b_i} = g(w^T x^i)$$

$\mathbf{E}$ is the unit matrix, and $\mathbf{E}^+$ is $\mathbf{E}$. So the solution is:

$$\mathbf{y} = \sum_i \mathbf{b_i} = \sum_i g(w^T x^i)$$

2.

a)

Iteration		μ_1	μ_2		x_1 class	x_2 class	x_3 class	x_4 class
1		0	3		1	2	2	2
2		1	11/3		1	1	2	2
3		1.5	4.5		1	1	2	2
4		1.5	4.5		1	1	2	2

b) Let $\pi_2 = 1 - \pi_1$, then we initialize the parameters and then iterate the following two steps:

E-step:

Compute w_{ij}, which is the probability that data point I is generated by Gaussian j.

$w_{ij} = \frac{\pi_j N(x_i|\mu_j,\Sigma_j)}{\sum_j \pi_j N(x_i|\mu_j,\Sigma_j)}$ let $n_j = \sum_i w_{ij}$

M-step:

Update the parameters:

$\boldsymbol{\mu}_j \leftarrow \sum_i w_{ij}\mathbf{x}_i/n_j$

$\boldsymbol{\Sigma}_j \leftarrow \sum_i w_{ij}(\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^T/n_j$

$\pi_i \leftarrow n_j/N$ (Where N is the total number of data points)

If we fix $\pi_1 = 0.5$ we just don't update π_i in the M-step above.

c)

Iter	π_1	μ_1	μ_2	σ_1^2	σ_2^2	w_{11}	w_{21}	w_{31}	w_{41}	w_{12}	w_{22}	w_{32}	w_{42}
1	0.5	0	3	1	1	0.8176	0.1824	0.0006	0.0000	0.1824	0.8176	0.9994	1.0000
2	0.25	1.1841	3.6058	0.1538	1.8157	0.8694	0.2111	0.0000	0.0000	0.1306	0.7889	1.0000	1.0000
3	0.27	1.1954	3.6679	0.1572	1.7157	0.8960	0.2596	0.0000	0.0000	0.1040	0.7404	1.0000	1.0000
4	0.29	1.2246	3.7213	0.1742	1.6442	0.9117	0.3548	0.0000	0.0000	0.0883	0.6452	1.0000	1.0000

3.

a) No.

The Bayes Network (a) and (b) both denotes $a \perp c \mid b$ so the joint distributions that can be represented by (a) and (b) are the same.

For (a) by defining CPTs of p(a) p(b|a) p(c|b), we know p(a,b,c)= p(a)p(b|a)p(c|b)

$$p(a,b) = \sum_c p(a,b,c) = \sum_c \mathrm{p(a)p(b|a)p(c|b)} = \mathrm{p(a)p(b|a)} \sum_c \mathrm{p(c|b)}$$
$$= p(a)p(b|a)$$
$$p(b,c) = \sum_a p(a,b,c) = \sum_a \mathrm{p(a)p(b|a)p(c|b)} = p(b)\mathrm{p(c|b)}$$
$$p(a,c) = \sum_b p(a,b,c) = \sum_b \mathrm{p(a)p(b|a)p(c|b)} = \mathrm{p(a)} \sum_b \mathrm{p(b|a)p(c|b)}$$

Same for (b), we get

$$p(\mathrm{a,b}) = \mathrm{p(a|b)p(b)}, \mathrm{p(b,c)} = \mathrm{p(b)p(c|b)}, \mathrm{p(a,c)} = \sum_{\mathrm{b}} \mathrm{p(a|b)p(b)p(c|b)}$$

So both Bayes Networks can represent p(a,b,c), p(a,b), p(b,c), and cannot represent p(a,c).

There is no joint distribution that can be represented using network in (a) that cannot be represented using network in (b).

b) Yes.

(a) denotes $a \perp c \mid b$ while (b) denotes $b \perp c \mid a$

Same as in 3.a) (a) network can represent p(a,b,c), p(a,b), p(b,c); while (b) network can represent p(a,b,c), p(a,b), p(a,c).

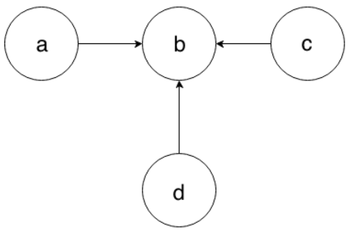
So there is a joint distribution (p(b,c)) that can be represented using network in (a) that cannot be represented using network in (b).

c) The joint distribution is

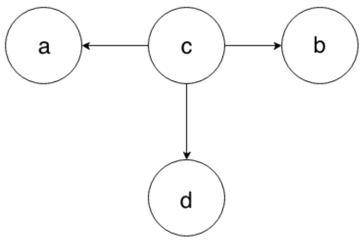
$$\mathrm{P(a,b,c,d) = P(a)P(b|a)P(c|a,d)P(d)}$$

d)

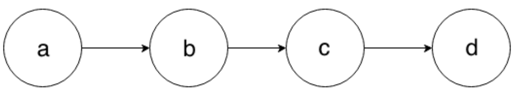
$a \not\perp c|b$



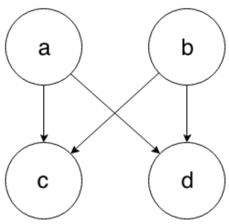
$a \perp b|c$



$a \not\perp b|\emptyset$

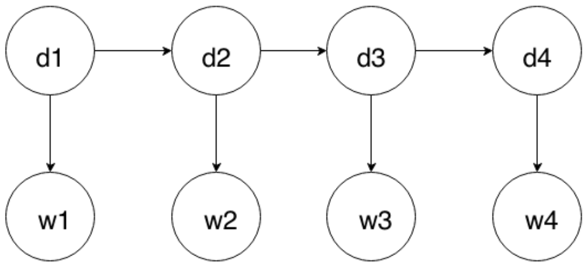


$a \not\perp b|c, d$



4.

a) The HMM Bayes network is:



dn for whether he dopped in year n; wn for whether he win in year n.

b)

$P(D_1|w_1) = \alpha P(w_1|D_1)P(D_1) = \alpha\langle 0.6,0.3\rangle\langle 0.5,0.5\rangle$

$= \langle 2/3,1/3\rangle \approx \langle 0.6667,0.3333\rangle$

$P(D_2) = \sum_{d_1} P(D_2|d_1)P(d_1|w_1) = \langle 0.6,0.4\rangle * 2/3 + \langle 0.1,0.9\rangle * 1/3$

$= \langle 13/30,17/30\rangle$

$P(D_2|w_{1:2}) = \alpha P(w_2|D_2)P(D_2) = \alpha\langle 0.6,0.3\rangle\langle 13/30,17/30\rangle$

$= \langle 26/43,17/43\rangle \approx \langle 0.6047,0.3953\rangle$

$P(D_3) = \sum_{d_2} P(D_3|d_2)P(d_2|w_{1:2}) = \langle 0.6,0.4\rangle * 26/43 + \langle 0.1,0.9\rangle * 17/43$

$= \langle 173/430,257/430\rangle$

$P(D_3|w_{1:3}) = \alpha P(w_3|D_3)P(D_3) = \alpha\langle 0.6,0.3\rangle\langle 173/430,257/430\rangle$

$= \langle 346/603,257/603\rangle \approx \langle 0.5738,0.4262\rangle$

$P(D_4) = \sum_{d_3} P(D_4|d_3)P(d_3|w_{1:3}) = \langle 0.6,0.4\rangle * 346/603 + \langle 0.1,0.9\rangle * 257/603$

$= \langle 2333/6030,3697/6030\rangle$

$P(D_4|w_{1:4}) = \alpha P(w_4|D_4)P(D_4) = \alpha\langle 0.6,0.3\rangle\langle 2333/6030,3697/6030\rangle$

$= \langle 4666/8363,3697/8363\rangle \approx \langle 0.5579,0.4421\rangle$

c)

First we compute backwards messages:

$P(w_4|D_4) = \langle 0.6,0.3\rangle$

$P(w_4|D_3) = \sum_{d_4} P(w_4|d_4)P(\,|d_4)P(d_4|D_3) = 0.6 * \langle 0.6,0.1\rangle + 0.3 * \langle 0.4,0.9\rangle$

$= \langle 0.48,0.33\rangle$

$P(w_{3:4}|D_2) = \sum_{d_3} P(w_3|d_3)P(w_4|d_3)P(d_3|D_2)$

$= 0.6 * 0.48 * \langle 0.6,0.1\rangle + 0.3 * 0.33 * \langle 0.4,0.9\rangle = \langle 0.2124,0.1179\rangle$

$P(w_{2:4}|D_1) = \sum_{d_2} P(w_2|d_2)P(w_{3:4}|d_2)P(d_2|D_1)$

$= 0.6 * 0.2124 * \langle 0.6,0.1\rangle + 0.3 * 0.1179 * \langle 0.4,0.9\rangle$

$= \langle 0.090612, 0.044577\rangle$

Then we combine these with the forwards messages computed previously and normalize:

$$P(D_1|w_{1:4}) = \alpha P(D_1|w_1)P(w_{2:4}|D_1) = \alpha\langle 2/3,1/3\rangle\, \langle 0.090612, 0.044577\rangle$$

$$\approx\ \langle 0.8026,0.1974\rangle$$

$$P(D_2|w_{1:4}) = \alpha P(D_2|w_{1:2})P(w_{3:4}|D_2) = \alpha\langle 26/43,17/43\rangle\langle 0.2124,0.1179\rangle$$

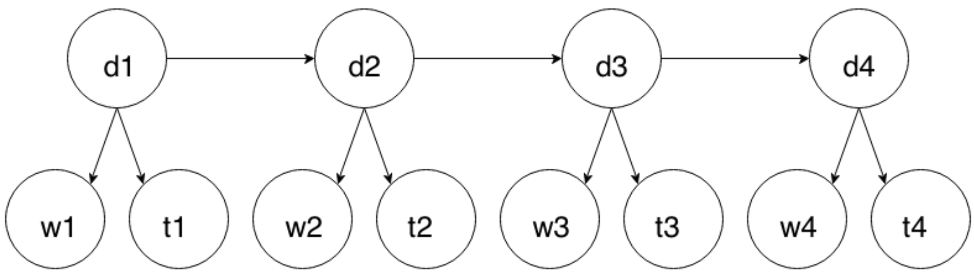
$$\approx \langle 0.7337,0.2663\rangle$$

$$P(D_3|w_{1:4}) = \alpha P(D_3|w_{1:3})P(w_4|D_3) = \alpha\langle 346/603,257/603\rangle\langle 0.48,0.33\rangle$$

$$\approx \langle 0.6620,0.3380\rangle$$

$$P(D_4|w_{1:4}) \approx \langle 0.5579,0.4421\rangle$$

d)
Given that drug tests are administered in each of the four years, we are adding new evidence to the HMM. Use tn to denote the test result of year n, we have:



5.

a) When in state a it can take action a->b or a->e. In other states, however, it can take only one action. So there are two distinct policies in this MDP:

$\pi(a \to b)$: Whenever it gets state a, take action a->b

$\pi(a \to e)$: Whenever it gets state a, take action a->e

b)

n denotes the nth time the MDP reach state a.

for $\pi(a \to b)$:

Since $R(a \longrightarrow b) = R(b \longrightarrow c) = R(c \longrightarrow d) = 0$

$$V(a_n) = \gamma V(b) = \gamma^2 V(c) = \gamma^3 V(d) = \gamma^3\big(R(d \longrightarrow a) + \gamma V(a_{n-1})\big)$$

$$= 5\gamma^3 + \gamma^4 V(a_{n-1})$$

For this array, we can calculate the nth element as:

$$V(a_n) = 5\gamma^3(1-\gamma^{4n})/(1-\gamma^4)$$

for $\pi(a \to e)$:

Since $R(a \longrightarrow e) = R(e \longrightarrow f) = R(f \longrightarrow g) = R(g \longrightarrow h) = R(h \longrightarrow i) = 0$

$$V(a_n) = \gamma V(e) = \gamma^2 V(f) = \gamma^3 V(g) = \gamma^4 V(h) = \gamma^5 V(i)$$

$$= \gamma^5\big(R(i \longrightarrow a) + \gamma V(a_{n-1})\big) = 6\gamma^5 + \gamma^6 V(a_{n-1})$$

For this array, we can calculate the nth element as:

$$V(a_n) = 6\gamma^5(1-\gamma^{6n})/(1-\gamma^6)$$

c)

Since $\gamma \in [0,1)$, when $n \longrightarrow \infty, \gamma^{4n} \longrightarrow 0, \gamma^{6n} \longrightarrow 0$

If we want the optimal policy take the agent through d. We need to have the

$$V\big(a_n, \pi(a \to b)\big) \geq V\big(a_n, \pi(a \to i)\big)$$

Which is: $5\gamma^3(1-\gamma^{4n})/(1-\gamma^4) \geq 6\gamma^5(1-\gamma^{6n})/(1-\gamma^6)$

$5\gamma^3/(1-\gamma^4) \geq 6\gamma^5/(1-\gamma^6)$

It is always true for $\gamma \in [0,1)$

So, for all $\gamma \in [0,1)$, the optimal policy takes the agent through d.

6.

a) The variables are Q_k $k \in \{1,2,3,4,5\}$, which denotes the row number of the Queen in column k.

Q_k can pick values in domain: $\{1,2,3,4,5\}$

The constraints is that $\forall$ i, j Q_i and Q_j are not attacking each other.

b)

remaining variables	possible values
Q_2	{3,4,5}
Q_3	{2,4,5}
Q_4	{2,3,5}
Q_5	{2,3,4}

c) Since all remaining variables have 3 remaining values, we assign $Q_2 = 3$ next.
After that:

remaining variables	possible values
Q_3	{5}
Q_4	{2}
Q_5	{2,4}

Q_3 and Q_4 has the minimum remaining values (1), we assign $Q_3 = 5$ next.
After that:

remaining variables	possible values
Q_4	{2}
Q_5	{2,4}

Q_4 has the minimum remaining values (1), we assign $Q_4 = 2$ next.
Then:

remaining variables	possible values
Q_5	{4}

assign $Q_5 = 4$, done.
Finally we have $Q_1 = 1, Q_2 = 3, Q_3 = 5, Q_4 = 2, Q_5 = 4$.

7.

a)

The state is a n-tuple, and n is the number of candidate positions (in this problem, there are 19 candidate positions among all 25 positions, so n is 19). Each element in the 19-tuple is a position (i,j) (i=1, 2, 3, 4; j=1, 2, 3, 4, 5, 6).

Since each element has 19 positions as candidate values, and each state contains 19 positions, so there are 19^{19}states in sum. In fact, it's just a theoretical upper bound because most of states are unreachable from the start state.

The start state S_{start} is a tuple contains all the candidate start positions, i.e.

$S_{start} = ((1,1), (1,2), (1,3), (1,5), (1,6), (2,3), (2,5), (2,6), (3,1), (3,2), (3,3), (3,5), (3,6), (4,1), (4,2), (4,3), (4,4), (4,5), (4,6))$

The successor function $f\,(x, a) = (f'(x_1, a), f'(x_2, a), \dots f'(x_{19}, a))$ where $x = (x_1, x_2, \dots, x_{19})$, and $f'(x_i)$ is a successor function for a position.

The goal state $S_{goal} = ((1,6), (1,6), \dots, (1,6),$ ((1,6) is the goal position, the upper right corner.).

b)

Use the heuristic function $\mathrm{h(S)} = \sum_{\mathrm{i}} \mathrm{M(S[i]\,, GOAL)}$

M(x,y) is the function of Manhattan distance. GOAL is (1,6) in this case. So the heuristic function is the sum of distances between each component position in current state and the goal position.

c)

Suppose $f^{-1}(x,a)$ is the inverse function of $f(x,a)$ for the variable $\mathbf{x}$ (a is treated as a constant value temporarily here). Then

$$P(x \mid a_1, \ldots, a_t) = \varepsilon P(x \mid a_1, \ldots, a_{t-1}) + (1-\varepsilon)(f^{-1}(x, a_t) \mid a_1, \ldots, a_{t-1})$$