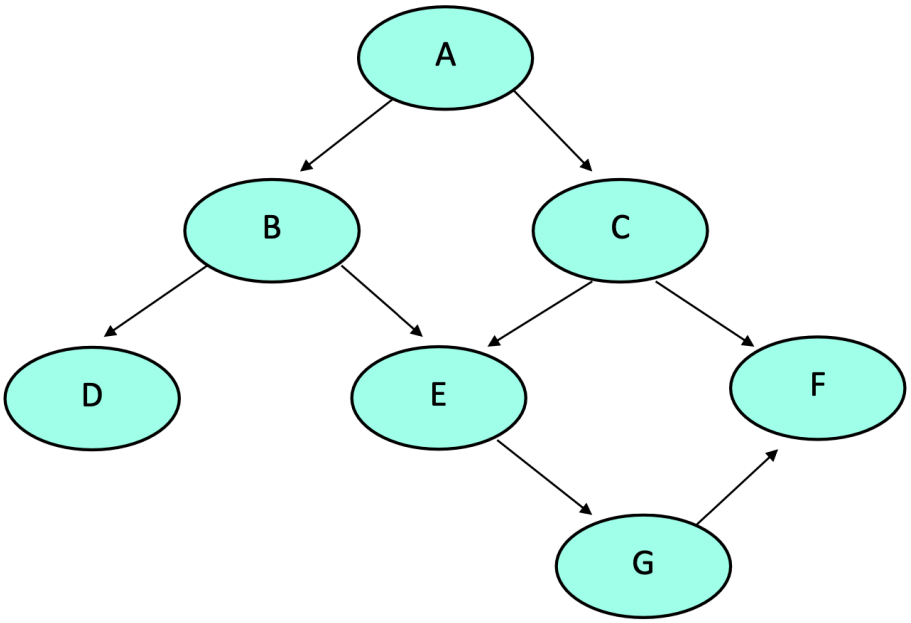
Question 1: In this Bayesian Network graph, analyze truthiness of each statement:



1. A⊥D

2. A⊥E |B

3. B⊥C|A

4. B⊥C|A,E

5. D⊥F|E

Answer 1:

1. Not correct. Based on “Cascade” independency rule, A and D are only independent if we are given B, and in this case B is not given. So A and D are dependent.

2. Correct. Exactly as 1, the same goes here. Based on “Cascade” independency rule, A and E are independent given B.

3. Correct. This case is correct based on “Common Parent” independency rule. B and C have common parent A and given A they are independent.

4. Not correct. In this case B and C have both “Common Parent” and “V-Structure” rules aapplied to them. Let’s start this way, because A is observed, B and C would typically be conditionally independent given A, but as E is also observed, E opens a path between B and C, potentially allowing information to "flow" between them. Therefore, B and C are conditionally **dependent** given A and E are observed and this is basically due to the “V-Structure” at E.

5. Not correct. D and F are actually conditionally dependent given E due to the “V-Structure” that we have at E. D and F do not have a direct path between them, but they are both influenced by E. And this is because the only path of influence

between D and F is through their parents B and C, respectively. And in between this path E acts as a "V-Structure". Hence, given E, D and F would be dependent.

Question 2: Suppose G is a Bayesian network and H is a Markov network on the set of variables X, such that the skeleton of the two graphs is the same. Under what conditions will I(G) = I(H)?

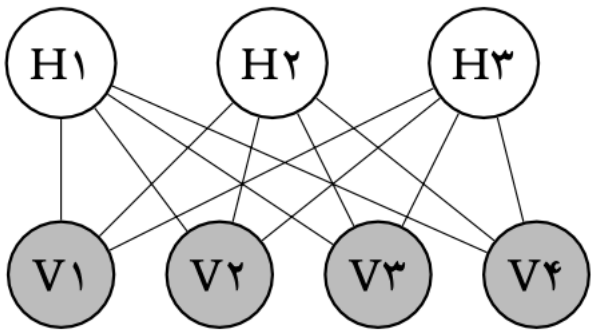
Answer 2: We have this definition as "Markov Equivalency", which essentially means that for G and H to be Markov Equivalent, they should have these checks:

1. They must have the same skeleton.
2. They must have the same set of "V-Structures".

Under this definition, and as the question states that both G and H share the same skeleton, then I(G) = I(H) if and only if G and H are Markov Equivalent, which means only if the V-Structures in G are the same as in H, added to the fact that they share the same skeleton.

Question 3: The restricted Boltzmann machine is a graphical model with a probability function as follows:

$$P(V,H) = \frac{1}{Z}\exp(\sum_i a_i h_i + \sum_j b_j v_j + \sum_{i,j} w_{ij} h_i v_j)$$



(a) Show that in this model, the conditional distribution of the hidden variables H given the observed variables V can be written as below:

$$P(H|V) = \prod_j P(H_j|V)$$

$$P(h_j = 1|V) = \sigma(a_j + \sum_i w_{ij} v_i)$$

(b) Find the factorized form of the distribution P(V|H). (If it is similar to part (a), detailed calculations are not necessary.)

(c) Obtain the distribution P(H). Can P(H) be factorized over the hidden variables H?

(d) In the graphical model of this distribution, are there any conditional independencies that cannot be derived from the probability function? Conversely, can all conditional independencies derived from the probability function be inferred from the graph?

Answer 3:

(a) To show the asked multiplication form of P(Hj|V) over all j(s), we need to start with the conditional independence structure of the RBM. In RBMs, hidden units (H) are conditionally independent given visible units (V), there are no direct connections between hidden units themselves in the graphical structure of an RBM. Therefore, the conditional distribution of each hidden unit (Hj) depends only on the visible units (V) and not on other hidden units. Therefore, we can conclude the given statement of $P(H|V)=\prod P(Hj|V)$ where each Hj is independent of other hidden units given V. Given the fact that each Hj is independent of other hidden units given V , this property allows us to factorize the joint probability of H given V into a product of individual conditional probabilities for each Hj.

In order to calculate $P(H|v)$, we use the form of conditional probs

$$P(H|v) = \frac{P(v,H)}{P(v)}, \quad P(v) = \sum_H P(v,H) = \sum_H \frac{1}{Z} \exp(-E(v,H))$$

marginal probability

Because in RBM each H_j is conditionally independent of other hidden units given the visible units v, we can factorize the joint conditional probability as:

$$P(H|v) = \prod_j P(H_j|v)$$

In energy function we have:

$$E(v,H) = -(a_j H_j + \sum_i w_{ij} H_j v_i) + (\text{other terms})$$

$$= -H_j(a_j + \sum w_{ij} v_i) + (\text{terms independent of } H_j)$$

Now for $P(H_j|v)$:

$\longrightarrow H_j = 1$

$$P(H_j = 1 \mid \mathcal{V}) = \frac{\exp(-E(\mathcal{V}, H_j = 1))}{\exp(-E(\mathcal{V}, H_j = 1)) + \exp(-E(\mathcal{V}, H_j = 0))} \quad (\star)$$

Now we can have:

$$E(\mathcal{V}, H_j = 1) = -\left(-a_j + \sum_i w_{ij} v_i\right) \quad \cdots\cdots$$

$$\exp(-E(\mathcal{V}, H_j = 1)) = \exp\left(a_j + \sum_i w_{ij} v_i\right) \quad (\star)$$

Using all the stars ($\star$) we can say:

$$P(H_j = 1 \mid \mathcal{V}) = \frac{\exp(a_j + \sum_i w_{ij} v_i)}{\exp(a_j + \sum_i w_{ij} v_i) + 1}$$

$$\underbrace{\qquad\qquad}_{n}$$

$$P(H_j = 1 \mid \mathcal{V}) = \frac{\exp(n)}{\exp(n) + 1} = \frac{1}{1 + \exp(-n)} \longrightarrow \boxed{\sigma(n) = \frac{1}{1 + \exp(-n)}} \longrightarrow = \sigma(n)$$

So $\longrightarrow$ $$P(H_j = 1 \mid \mathcal{V}) = \sigma\left(a_j + \sum_i w_{ij} v_i\right)$$

(b) Same as (a) we can say:

$$P(\mathcal{V} \mid H) = \frac{P(\mathcal{V}, H)}{P(H)}$$

As the visible units ($\mathcal{V}$) are conditionally independent of each other given (H). So, we can write as below:

$$P(\mathcal{V} \mid H) = \prod_j P(v_j \mid H)$$

And for each $P(v_j|H)$ we can have below as before:

$$P(v_j=1|H) = \sigma\left(b_j + \sum_i w_{ij} h_i\right)$$

We mean that the probability that a visible unit v_j is activated ($v_j=1$) depends on H plus a bias, this is pretty much the same as previous parts in v and H are swapped.

So we can combine everything and say:

$$P(v|H) = \prod_j P(v_j|H), \begin{cases} P(v_j=1|H) = \sigma(b_j + \sum_i w_{ij} h_i) \\ P(v_j=0|H) = 1 - \sigma(b_j + \sum_i w_{ij} h_i) \end{cases}$$

(c) See below:

For $P(H)$, we should marginalize over all v: $P(H) = \sum_v P(v,H)$

From the statement, $P(H) = \frac{1}{Z} \exp\left(\sum_i a_i h_i\right) \sum_v \exp\left(\sum_j b_j v_j + \sum_{i,j} w_{ij} h_i v_j\right)$

Because of RBM:

$$\sum_v \exp\left(\sum_j b_j v_j + \sum_{i,j} w_{ij} h_i v_j\right) = \prod_j \left(\sum_{v_j} \exp\left(b_j v_j + \sum_i w_{ij} h_i v_j\right)\right)$$

Also as $v_j \in \{0,1\}$:

$$\sum_{v_j} \exp\left(b_j v_j + \sum_i w_{ij} h_i v_j\right) = 1 + \exp\left(b_j + \sum_i w_{ij} h_i\right)$$

So, from all these we can write $P(H)$ as:

$$P(H) = \frac{1}{Z} \exp\left(\sum_i a_i h_i\right) \prod_j \left(1 + \exp\left(b_j + \sum_i w_{ij} h_i\right)\right) \quad \checkmark$$

For the factorization: $P(H) \stackrel{?}{=} \prod_j P(H_j)$

→ Because inside the $\prod$, we have $\left(\sum_i w_{ij} h_i\right)$ and c_j, involves all hidden units H_i, we have dependencies between the hidden units in $P(H)$. Hence, $P(H) \neq \prod_j P(H_j)$ ✗

(d)

From the graphical model, we get the following independencies:

- Because no direct connections exist between hidden units in the graphetween hidden units, given the visible units, we have independencies as below:

$$P(Hj,Hk|V)=P(Hj|V)\ P(Hj|V)\ ;\ j\neq k$$

- Same goes for visible units and we have:

$$P(Vj,Vk|H)=P(Vj|H)\ P(Vj|H)\ ;\ j\neq k$$

From the probability function we get no additional conditional independencies beyond the graph. Because, the RBM's bipartite structure ensures that the only conditional independencies are those directly implied by the absence of intra-layer connections. Also there are no hidden conditional independencies in P(V,H) that are not already captured by the graph's structure.

For the vice-versa, all conditional independencies are captured by the graph. Because, any conditional independence present in the probability distribution P(V,H) can be inferred from the graphical model. The graph fully represents the conditional independence structure of P(V,H).

Question 4: By reading sections 1-4 of this article, briefly explain how Structural Causal Model (SCM) is used to enhance the interpretability of deep models.

Answer 4: SCMs can enhance the interpretability of deep learning models in several ways:

Modeling Deep Networks as SCMs: SCMs map out the causal linkages between various components (such as neurons and layers) and the final outputs in order to describe the internal structure of deep neural networks. This causal graph illustrates how modifications to one aspect of the model have a causal impact on other aspects. It is usually a Directed Acyclic Graph (DAG).

One can calculate the Average Causal Effect (ACE) of each neural network component on the model's predictions by treating them as variables inside an SCM. This helps identify which parts of the network are most influential in decision-making.

SCMs enable the creation of counterfactual scenarios to understand how these alterations might affect output of the model. For example, in a credit application scenario, SCMs can answer questions like, "What if the applicant's income was higher? Would the decision change?"

SCMs help manage multiple valid counterfactual explanations by providing a structured approach to evaluate and select the most meaningful ones, enhancing the clarity and usefulness of explanations.

SCMs facilitate the detection and adjustment for confounding variables that may introduce biases into the model's decisions. By explicitly modeling these

confounders, SCMs ensure that explanations account for potential sources of inequality, thereby promoting fairer outcomes.

The use of SCMs allows for the definition and measurement of fairness based on causal effects, which ensures that decisions are not only statistically unbiased but also causally justifiable.

SCMs provide a method to quantify the contribution of individual model components (such as specific neurons or layers) to the final decision. This understanding helps diagnosing and improving model behavior.

By extracting and utilizing human-interpretable concepts (e.g., features like "eyes" and "ears" in image recognition models) within the SCM framework, explanations become more intuitive and aligned with human reasoning.

SCMs also serve as a tool to verify that the causal relationships inferred from data align with the model's decision-making processes. This verification step is crucial for building trust in model explanations, especially in high-stakes applications like healthcare.

In overall, integrating SCMs into deep learning frameworks provides a robust and systematic approach to interpretability. SCMs offer deeper insights into how and why models make specific decisions. This enhanced interpretability is essential for ensuring the reliability, fairness, and transparency of machine learning models in critical applications.

Question 5: In the class, we have analyzed the parameters of HMM model in the case that all random variables of model are observed. Now consider the case, where the model should only train from sentences and no hidden variables are present. In this case, to learn the variables, we must use the Expectation-Maximization algorithm. Obtain the steps of the algorithm for this model. To achieve this, you can use this source as a guide. (Note that copying the text from the source to this exercise will not earn you any points, and you will need to rephrase the expressions after studying them yourself).

Answer 5: So the problem setup formally would be:

Problem Setup:

The algorithm is given:

- A sequence of observations $X=\{x1, x2, \dots, xT\}$
- HMM parameters to estimate:
  - Transition probabilities A
  - Initial state probabilities π
  - Emission probabilities Φ

By iteratively estimating the parameters using the EM algorithm we aim to maximize the likelihood of the observations.

Algorithm:

1. Initialization
    1. Initialize parameters Φ, π, A
        1. π[i]: Probability of starting in state
        2. A[i][j]: Probability of transitioning from state i to state j
        3. Φ[i][x]: Probability of emitting observation x from state I
2. E-Step
    1. Forward Algorithm:
        1. Calculate the probability of observing the sequence up to time t ending in state i, using the forward variable $\alpha_t(i)$

$$\alpha_1(i) = \pi[i] \cdot \Phi[i][x_1]$$

$$\alpha_t(i) = \sum_j \alpha_{t-1}(j) \cdot A[j][i] \cdot \Phi[i][x_t]$$

    2. Backward Algorithm:
        1. Calculate the probability of observing the remaining sequence from t+1 given that we start in state i at time t, using backward variable $\beta_t(i)$

$$\beta_T(i) = 1$$

$$\beta_t(i) = \sum_j A[i][j] \cdot \Phi[j][x_{t+1}] \cdot \beta_{t+1}(j)$$

    3. Calculate γ and ξ:
        1. $\gamma_t(i)$: Probability of being in state i at time t:

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{\sum_k \alpha_t(k) \cdot \beta_t(k)}$$

        2. $\xi_t(i, j)$: Probability of transitioning from state i at time t to state j at time t+1:

$$\xi_t(i, j) = \frac{\alpha_t(i) \cdot A[i][j] \cdot \Phi[j][x_{t+1}] \cdot \beta_{t+1}(j)}{\sum_{i,j} \alpha_t(i) \cdot A[i][j] \cdot \Phi[j][x_{t+1}] \cdot \beta_{t+1}(j)}$$

3. M-Step
    1. We update the parameters Φ, π, A using the values of γ and ξ:
        1. Update of π : The initial state distribution

$$\pi[i] = \gamma_1(i)$$

        2. Update of A: Transition probabilities between states

$$A[i][j] = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

        3. Update of Φ: Emission probabilities for each observation given the state

$$\Phi[i][x] = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot \mathbb{I}(x_t = x)}{\sum_{t=1}^{T} \gamma_t(i)}$$

Where $\mathbb{I}(x_t = x)$ is an indicator function that is 1 if $(x_t = x)$, and 0 otherwise.

4. Iteration and Convergence
    1. Repeat the E-Step and M-Step until the parameters converge. (i.e., the change in likelihood of the observed data between iterations is below a threshold.)

Pseudocode algorithm:

```
# Initializing
initialize_parameters(pi, A, Phi)
# E-Step
alpha = forward_algorithm(X, pi, A, Phi)
beta = backward_algorithm(X, A, Phi)
gamma = compute_gamma(alpha, beta)
xi = compute_xi(alpha, beta, A, Phi, X)
# M-Step
pi = gamma[:, 0]
for i in states:
    for j in states:
        A[i][j] = sum(xi[t][i][j] for t in range(T-1)) / sum(gamma[t][i] for t in range(T-1))
for i in states:
    for x in observations:
        Phi[i][x] = sum(gamma[t][i] for t in range(T) if X[t] == x) / sum(gamma[t][i] for t in range(T))
# EM Iteration
while not_converged:
    alpha, beta = forward_algorithm(X, pi, A, Phi), backward_algorithm(X, A, Phi)
    gamma, xi = compute_gamma(alpha, beta), compute_xi(alpha, beta, A, Phi, X)
    pi, A, Phi = update_parameters(gamma, xi, X)
    check_convergence()
```

Question 6: We want to use Naïve Bayes to check whether a received email is work-related or not. Using the data from the chart below, examine each email and check if they are work related or not. What is the difference between this method and Discriminative models?

Email	Words used	Work related?
1	schedule – event – schedule	Yes
2	schedule – schedule – meeting	Yes
3	event – deadline	Yes
4	movie – lunch – schedule	No
5	schedule – movie – lunch – schedule	?

Answer 6:

First we should do some preprocessing:

1. Tokenization: Split the "words used" field into individual words.
2. Vocabulary: Identify all unique words across all emails.
3. Frequency: Count how often each word appears in emails labeled "Yes" and “No”.

For vocabulary: Unique words are:
- schedule
- event
- meeting
- deadline
- movie
- lunch

We should then compute prior probabilities, which are the probabilities of each class (Work Related: Yes/No) before considering any words.

$$P(\text{Yes}) = \frac{\text{Number of "Yes" emails}}{\text{Total number of emails with known labels}} = \frac{3}{4} = 0.75$$

$$P(\text{No}) = \frac{\text{Number of "No" emails}}{\text{Total number of emails with known labels}} = \frac{1}{4} = 0.25$$

Now, to handle words that may not appear in a particular class, we use Laplace smoothing (adding 1 to each count).

“Yes” Class:
- Total words in “Yes” emails:
  - Email 1: schedule (2), event (1)
  - Email 2: schedule (2), meeting (1)
  - Email 3: event (1), deadline (1)
  - Total = 2 + 1 + 2 + 1 + 1 + 1 = 8
- Words counts:
  - schedule: 4
  - event: 2
  - meeting: 1
  - deadline: 1
  - movie: 0
  - lunch: 0

Adding Laplace Smoothing:

$$P(\text{word}\,|\,\text{Yes}) = \frac{\text{Count of word in "Yes" emails} + 1}{\text{Total words in "Yes" emails} + \text{Vocabulary size}} = \frac{\text{Count} + 1}{14}$$

Calculating for schedule:

$$P(\text{schedule}\,|\,\text{Yes}) = \frac{4+1}{14} = \frac{5}{14} \approx 0.357$$

Doing the same for all words we get:
- schedule: (4 + 1) / 14 = 5/14 ≈ 0.357
- event: (2 + 1) / 14 = 3/14 ≈ 0.214
- meeting: (1 + 1) / 14 = 2/14 ≈ 0.143
- deadline: (1 + 1) / 14 = 2/14 ≈ 0.143
- movie: (0 + 1) / 14 = 1/14 ≈ 0.071
- lunch: (0 + 1) / 14 = 1/14 ≈ 0.071

"No" Class:
- Total words in "No" emails:
  - Email 4: movie (1), lunch (1), schedule (1)
  - Total = 1 + 1 + 1 = 3
- Words counts:
  - schedule: 1
  - event: 0
  - meeting: 0
  - deadline: 0
  - movie: 1
  - lunch: 1

Adding Laplace Smoothing:

$$P(\text{word}\,|\,\text{No}) = \frac{\text{Count} + 1}{9}$$

Calculating for words:
- schedule: (1 + 1) / 9 = 2/9 ≈ 0.222
- event: (0 + 1) / 9 = 1/9 ≈ 0.111
- meeting: (0 + 1) / 9 = 1/9 ≈ 0.111
- deadline: (0 + 1) / 9 = 1/9 ≈ 0.111
- movie: (1 + 1) / 9 = 2/9 ≈ 0.222
- lunch: (1 + 1) / 9 = 2/9 ≈ 0.222

Well now we can classify Email 5:

- Words used: schedule, movie, lunch, schedule
- Calculating the probability for each class

Well Naïve Bayes classification general formula is:

$$P(C \mid \mathbf{w}) \propto P(C) \times \prod_{i=1}^{n} P(w_i \mid C)$$

Using the Naïve Bayes assumption which is conditional independence: For Email 5 with words: schedule, movie, lunch, schedule:

$$P(\text{Yes} \mid \text{Words}) \propto P(\text{Yes}) \times P(\text{schedule} \mid \text{Yes})^2 \times P(\text{movie} \mid \text{Yes}) \times P(\text{lunch} \mid \text{Yes})$$
$$P(\text{No} \mid \text{Words}) \propto P(\text{No}) \times P(\text{schedule} \mid \text{No})^2 \times P(\text{movie} \mid \text{No}) \times P(\text{lunch} \mid \text{No})$$

To calculate the actual probabilities, we first calculate then normalize the proportional probabilities calculated, as below:

$$P(\text{Yes} \mid \text{Words}) = \frac{P(\text{Yes}) \times P(\text{schedule} \mid \text{Yes})^2 \times P(\text{movie} \mid \text{Yes}) \times P(\text{lunch} \mid \text{Yes})}{P(\text{Yes} \mid \text{Words}) + P(\text{No} \mid \text{Words})} \approx 0.442$$

$$P(\text{No} \mid \text{Words}) = \frac{P(\text{No}) \times P(\text{schedule} \mid \text{No})^2 \times P(\text{movie} \mid \text{No}) \times P(\text{lunch} \mid \text{No})}{P(\text{Yes} \mid \text{Words}) + P(\text{No} \mid \text{Words})} \approx 0.558$$

So, we have:
- P(Yes | Words) ≈ 44.2%
- P(No | Words) ≈ 55.8%

And since probability of No is greater than Yes, Email 5 should be classified as Not work related.

Now to check the difference between Naïve Bayes and Discriminative Models (Some information is borrowed from lecture 2 of Dr. Stefano Ermon's DGM course on YouTube):

Naïve Bayes (and generative models basically) aim to model the joint probability distribution P(X,Y), where X represents the features and Y the class labels. They learn how the data is generated in order to categorize new instances.

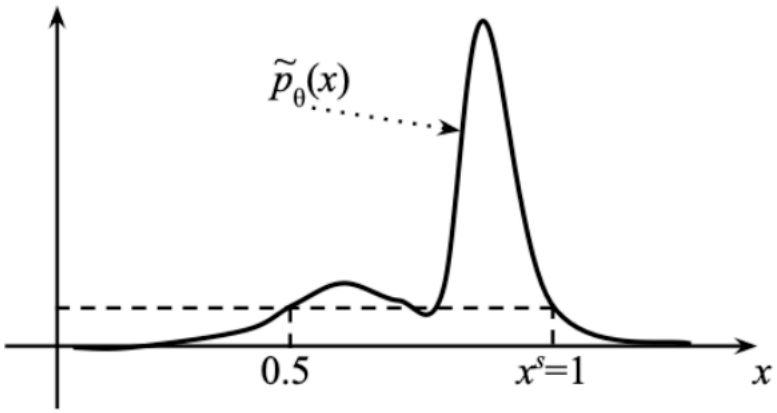
Discriminative models (e.g., Logistic Regression) focus on modeling the conditional probability P(Y|X) directly. They aim to find a boundary that separates classes without necessarily understanding how the data was generated.

Now let's check characteristics of each one, based on category and compare them:

- What they model:
  - Generative models: model the joint prob P(X,Y)
    - They model the "Data Generation Process"
    - They can generate new data instances resembling the training data.
  - Discriminative Models: model the conditional prob P(Y|X)
    - They model the "Decision Boundary"

    - They learn to distinguish between classes by finding a function that maps input features to class labels.
- What they assume:
    - Generative models: assume feature independence
    - Discriminative Models: fewer assumptions about feature relationships
- Can they generate data:
    - Generative models: can generate new data instances
    - Discriminative Models: cannot generate new data
- Performance:
    - Generative models: can perform worse if assumptions are wrong
    - Discriminative Models: usually achieve higher classification accuracy
- Flexibility:
    - Generative models: less flexible due to strong assumptions
    - Discriminative Models: more flexible in capturing complex patterns
- So overally:
    - Generative models: suitable for scenarios with missing data
    - Discriminative Models: are preferred for pure classification tasks

Question 7: The diagram below shows the non-normalized distribution $\tilde{p}_\theta(x)$:



We want to run a step of the Metropolis-Hastings algorithm starting from the point $x^s = 1$. If the proposed distribution is exponential, answer the following questions:

$$q(X \mid x^s) = \text{Exp}(X \mid \lambda = x^s + 1)$$

(a) If the result of sampling from the proposed distribution is the mean of this distribution, what is x^p in that case?

(b) Calculate the value of $\alpha = \dfrac{\tilde{p}(x^p)}{\tilde{p}(x^s)}$.

(c) Compute the Hastings Correction, which is $\dfrac{q(x^s \mid x^p)}{q(x^p \mid x^s)}$.

(d) Compare the following expressions and explain the impact of the Hastings Correction on the proposed point for sampling and the proposed distribution:

$$\alpha_1 = \frac{\tilde{p}(x^p)}{\tilde{p}(x^s)}, \quad \alpha_2 = \frac{\tilde{p}(x^p)q(x^s \mid x^p)}{\tilde{p}(x^s)q(x^p \mid x^s)}$$

Answer 7:

(a) We want to sample from the exponential distribution provided with the mean of it. So we need to find the mean of this distribution. As we we know the mean in exponential distribution is calculated from dividing 1 by the rate parameter λ. So:

$$\lambda = x^s + 1 = 2$$
$$x^p = \frac{1}{2}$$

(b) to find the ratio α from the diagram we see that $\tilde{p}(x^p) = \tilde{p}(x^s)$ so:

$$\alpha = \frac{\tilde{p}(x^p)}{\tilde{p}(x^s)} = 1$$

Hence, the proposed move to x^p is equally likely to be accepted based on the distribution alone.

(c) To compute the Hastings Correction, we will use the formular provided. If we expand the formula we get:

$$HastingCorrection = \frac{q(x^s \mid x^p)}{q(x^p \mid x^s)}$$

$$q(X \mid x^s) = Exp(X \mid \lambda = x^s + 1) = Exp(X \mid \lambda = 2)$$
$$q(X \mid x^p) = Exp(X \mid \lambda = x^p + 1) = Exp(X \mid \lambda = 1.5)$$

Using definition of exponential probability density function:

$$q(X \mid \lambda) = \lambda e^{-\lambda X}$$

So we get:

$$q(x^s \mid x^p) = 1.5e^{-1.5}$$
$$q(x^p \mid x^s) = 2e^{-1}$$

Putting back in the formular we get:

$$HastingCorrection = \frac{q(x^s \mid x^p)}{q(x^p \mid x^s)} = \frac{1.5e^{-1.5}}{2e^{-1}} = \frac{3}{4}e^{-0.5}$$

(d) We are going to explain the expressions below:

$$\alpha_1 = \frac{\tilde{p}(x^p)}{\tilde{p}(x^s)} = 1, \quad \alpha_2 = \frac{\tilde{p}(x^p)q(x^s \mid x^p)}{\tilde{p}(x^s)q(x^p \mid x^s)} = \frac{3}{4}e^{-0.5}$$

Hastings correction modifies the acceptance ratio to account for the asymmetry of the proposal distribution, but since $\alpha_1 = 1$, α_2 directly reflects this hastings correction. If α_1 is the acceptance ratio without the Hastings correction, then α_2 is including this correction to ensure balance and correct sampling. This ensures that even if the distribution values are equal, the movement between points is adjusted to maintain detailed balance in this algorithm.
As $\alpha_1 = 1$, the Hastings correction ratio will determine whether α_2 is also equal to 1, greater than 1, or less than 1.

Therefore in overall we can say,
- If Hastings Correction > 1 it increases the acceptance probability.
- If Hastings Correction < 1 it decreases the acceptance probability, showing that moving from x^s to x^p is less probable compared to the reverse move.

Question 8: Consider a random variable X_i for $i \in \{1,...,N\}$ such that $X_i \sim \text{Normal}(X_i; \mu, \tau^{-1})$ are i.i.d. Assume that the parameters of this distribution are indeterministic, such that:

$$\mu \sim \text{Normal}(\mu; \mu_0, (\tau\tau_0)^{-1}), \tau \sim \text{Gamma}(\tau; \alpha_0, \beta_0)$$

Where $\alpha_0, \beta_0, \mu_0, \tau_0$ are given constants. We want to find the posterior distribution $p(\mu, \tau \mid x)$ using variational inference. According to the Mean-Field method, take the approximate of the posterior probability as $q(\mu, \tau) = q(\mu)q(\tau)$ and calculate $q(\mu)$ and $q(\tau)$.
Hint: $q(\mu)$ is normal distribution and $q(\tau)$ is gamma distribution.

Answer 8:
In the Mean-Field approach, we go through iterations and in each iteration update each q distribution while keeping the others fixed.

1. Initializing:
    1. $q(\mu) = \text{Normal}(\mu \mid m, s^2)$
    2. $q(\tau) = \text{Gamma}(\tau \mid a, b)$
2. Updating:
    1. Updating $q(\mu)$:

        To get optimum factors, we should calculate as below:

$$\log q^*(\mu) \propto \mathbb{E}_{q(\tau)}[\log p(\mu, \tau, x)]$$

        Now we can find the new parameters of $q(\mu)$ as a function of the current parameters of $q(\tau)$.

        Update equations for parameters m and s^2:

$$s^2 = \left(\tau_0 \mathbb{E}[\tau] + N\mathbb{E}[\tau]\right)^{-1}$$
$$m = s^2\left(\tau_0 \mathbb{E}[\tau]\mu_0 + N\mathbb{E}[\tau]\bar{X}\right)$$

        Where $\mathbb{E}[\tau] = \frac{a}{b}$ from $q(\tau)$.

    2. Updating $q(\tau)$

        The same goes here,

$$\log q^*(\tau) \propto \mathbb{E}_{q(\mu)}[\log p(\mu, \tau, x)]$$

        And contributions from the prior τ and likelihood form:

$$a' = \alpha_0 + \frac{N+1}{2}$$
$$b' = \beta_0 + \frac{1}{2}\sum_i (X_i - m)^2 + \frac{\tau_0}{2}(m - \mu_0)^2$$

        For the:

$$q(\tau) = \text{Gamma}(\tau \mid a', b')$$

3. We iteratively do these updates until the parameters m, s, a, b converge. This process gradually refines the approximations to converge towards the true posterior. [proof in class slides]