COMP 560: Artificial Intelligence
March 17, 2025

Assignment 2
Due Date: April 10, 2025

Please share your Google Colab links (make sure permissions are set to everyone with the link) or upload the ipynb files a s an assignment solution. On Google Colab, you can download the i pynb by clicking File / Download / Download .ipynb.

Question 1. Probability Basics (25 points)

a). Conditionals and Marginals (10 points)

A travel group is considering whether they want to hike based on the current weather conditions. There are Boolean variables in this situation: rain, cloud and hike. Below is the table for the joint probability distribution *p*(*rain, cloud, hike*).

Note: ¬ means the negation (“not”).

	cloud		¬ cloud	
	rain	¬ rain	rain	¬ rain
hike	0.05	0.19	0.05	0.19
¬ hike	0.18	0.10	0.14	0.10

Find the following probabilities:

i) (2 points) P(rain)

We can sum all probabilities where p(rain) is true regardless of what the cloud and hike is
p(rain)= p(rain, clouds, hike) + p(rain, cloud, not hike) + p(rain, not cloud, not hike) + p(rain, not clout, hike) = 0.05+ 0.18+ 0.14+ 0.05= 0.42

ii) (4 points) *P*(*cloud* | ¬*hike*)

p(cloud| *¬hike) = p(cloud, ¬hike)/ p(¬hike)*
p(cloud, ¬hike)= 0.18+0.10= 0.28
p(¬hike)= 0.18+0.10+0.14+0.10= 0.52
p(cloud| *¬hike) = 0.28/0.52 = 0.538*

ii) (4 points) *P*(*rain* | *cloud, hike*)

p(rain, cloud, hike)/ p(cloud, hike)
p(rain, cloud, hike)= 0.05
p(cloud, hike)= 0.05+ 0.19=0.24
P(*rain* | *cloud, hike*) = 0.208

b). Bayes Rulezz (5 points)

The wonderful Prof. Srivastava has developed a test to detect if a person has a virus. This test gives a positive result with a probability of 85% if it is present. Unfortunately, it also gives a positive result with probability 5% (the false positive rate) even if it is not present. It is known that 2% of the population has the virus. Assume a person used the test, and the test gave a negative result. What is the probability the person does not have the virus?

p(V) = 0.02
p(¬v) = 1- 0.02 =0.98
p(neg|v) =1-0.85=0.15
p(neg| ¬v)= 1-0.05=0.95

P(neg) =p(neg|v) p(v)+ p(neg|¬v) p(¬v)= 0.15(0.02)+0.95(0.98)= 0.003+0.931=0.934
*p(¬v|neg) = 0.95*0.98/0.934 = 0.9968*

*p(***c). Great Expectations (10 points)**

i). (4 points) Assume the time of a phone call can be modelled by the probability density function $p(x) = c/t^2$ for $1 \le t \le 40$ and $p(t) = 0$ otherwise. What is c? Find the average length of a phone call. **ii). (6 points)** There is a phone plan with a phone price of \$(0.5 − 0.002t) per minute where t is the time since the beginning of the call. What would the average cost be if the distribution of phone call time is the same as part i)?

NOTE: the phone time is assumed to be continuous. For the cost of a phone call of length t you should integrate $\int_0^t f(t_1)\, dt_1$ where $f(t_1)$ is the price at minute t_1.

We need to find two things, find c and average Length also Average cost

a) $P(t) = \frac{c}{t^2}$, for $1 \le t \le 40$

$\int_1^{40} \frac{c}{t^2} dt = 1 \quad c\left[-\frac{1}{t}\right]_1^{40} = 1 \quad c\left(-\frac{1}{40} + 1\right) = 1 \quad c = \frac{39}{40} = 1$

$c = \frac{40}{39}$

Average time $E[t] = \int_1^{40} t \cdot \frac{c}{t^2} dt = c \int_1^{40} \frac{1}{t} dt = c \cdot \ln(40)$

$= \frac{40}{39} \cdot \ln(40) \approx \frac{40}{39} \cdot 3.6889 = 3.784$

b) Average cost : $f(t) = 0.5 - 0.002t$

$E[cost] = \int_1^{40} \left(\int_0^t (0.5 - 0.002x)\, dx \right) \cdot \frac{c}{t^2} dt$

$\int_0^t (0.5 - 0.002x)\, dx = [0.5x - 0.001x^2]_0^t = 0.5t - 0.001t^2$

$E[cost] = \int_1^{40} (0.5t - 0.001t^2) \cdot \frac{c}{t^2} dt = c \int_1^{40} \left(\frac{0.5t - 0.001t^2}{t^2} \right) dt$

$= c \int_1^{40} \left(\frac{0.5}{t} - 0001 \right) dt$

$= c\, [0.5 \ln(40) - 0.001) - (40 - 1) = \frac{40}{39} [0.5 \cdot 3.6889 - 0.039$

$= \frac{40}{39} (1.8445 - 0.039) = \frac{40}{39} \cdot 1.8055 = 1.852$

$E[cost] = \$1.85$

Question 2. Bayesian Networks (25 points)

a). Designing a Bayes Net (6 points) You are building a safety detection for a robot in real-time and you want to model this. There are multiple variables that you need to think about:

- E: the robot may be in danger.
  - C: the robot may crash
  - O: the robot may be moving towards an obstacle.
  - B: the robot's battery may be low
  - T: the robot's temperature may be too high or too low.
  - M: the internal system may be malfunctioning

B can influence E or M. M can influence S. S and O can influence C. E can be influenced by B, C, T. Draw a Bayesian network that models this situation.

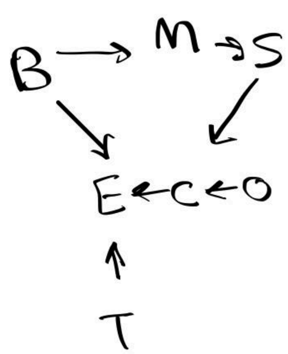
We would need to create a DAG

Since B affects both M and E

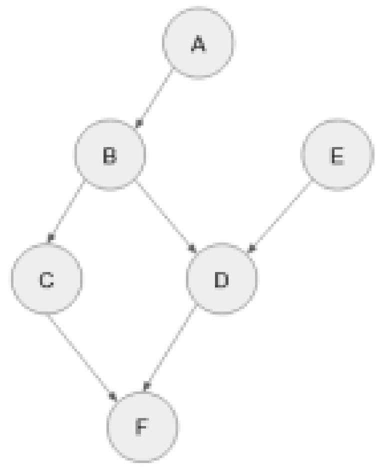
M affects S

S and O together affect C

C,T and B all affect E



b). Declaring Independence (6 points) Consider the following Bayesian network. Which of the following independence statements are guaranteed from the structure of the network? Briefly explain why.



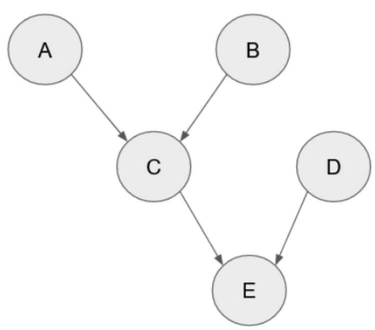
i) *A ⊥ F | C* *A is not independent of F given C*
Conditioning on C does not block the path A → B → D → F (since D is not conditioned on and is a child of B).
So, A is still connected to F even given C.

ii) *A ⊥ F | C, D* *A is independent of F given C and D.*
The first path A → B → C → F is blocked because C is conditioned on.

The second path A → B → D → F is also blocked because D is conditioned on.

iii) *A ⊥ E* *A is independent of E*
A → B → D ← E we are not conditioning on anything, and D is a collider, so the path is blocked.

c). Exact(ing) Inference (13 points)
Consider the following Bayesian network with the associated conditional probability tables:



A	$P(A)$
T	0.80
F	0.20

B	$P(B)$
T	0.30
F	0.70

D	$P(D)$
T	0.90
F	0.10

A	B	C	$P(C \mid A, B)$
F	F	F	0.45
F	F	T	0.55
F	T	F	0.10
F	T	T	0.90
T	F	F	0.35
T	F	T	0.65
T	T	F	0.30
T	T	T	0.70

C	D	E	$P(E \mid C, D)$
F	F	F	0.55
F	F	T	0.45
F	T	F	0.10
F	T	T	0.90
T	F	F	0.25
T	F	T	0.75
T	T	F	0.70
T	T	T	0.30

TABLE 1. Conditional probability tables for above Bayesian network

i) (3 points) Write *P*(*A, B, C, D, E*) as a product of conditional probabilities Use Variable Elimination to calculate the following probabilities:

A,B,D are independent root nodes → P(A)P(B)P(D)

C depends on A,B → P(C∣A,B)

E depends on C,D → P(E∣C,D)

P(*A, B, C, D, E*)= P(A)*P(B)*P(D)** P(C∣A,B)* P(E∣C,D)

3

ii) (4 points) *P*(*A, C*) 0.532
iii) (6 points) *P*(*A, E, ¬D*) 0.05464

Question 3. Wizardry with Weights (20 points)

Importance Sampling is a powerful technique for estimating properties of complex distributions. In this question, we will assume that we are only able to sample from a Gaussian distribution with a mean of 5 and a variance of 2. Starting from this, we will use Importance Sampling to approximate three target distributions of varying complexity: a different Gaussian distribution, a mixture of Gaussians, and a mixture of uniform distributions.

Target 1: Gaussian Distribution First, we will try to approximate a different Gaussian distribution (with a mean of 2, and variance 1) with the following probability density function:

$$f(x) = \frac{\exp\left(-\frac{1}{2}(x-2)^2\right)}{\sqrt{2\pi}}.$$

Target 2: Mixture of Gaussians Next, we will explore approximating a target distribution that is a mixture of two Gaussian distributions, with the following probability density function:

$$f(x) = 0.3 \cdot \frac{\exp\left(-\frac{1}{2}(x-2)^2\right)}{\sqrt{2\pi}} + 0.7 \cdot \frac{\exp\left(-\frac{1}{2}(x-10)^2\right)}{\sqrt{2\pi}}.$$

Target 3: Mixture of Uniform Distributions Finally, we will consider approximating a discontinuous target distribution that is a mixture of two uniform distributions, defined by the following probability density function:

$$f(x) = \begin{cases} 0.1 & \text{if } 2 \le x \le 5, \\ 0.233333 & \text{if } 10 \le x \le 13, \\ 0 & \text{otherwise.} \end{cases}$$

a) (3 points) Implement importance sampling to approximate each target distribution. Most of the code for doing this is already provided in the Google Colab link at: https://colab.research.google.com/drive/1Y5USrgTkaPWhP4MU2C_uGnZBeSHeeIVj

You will need to complete the calculate importance weights function to compute the adjusted weight for each sample.

b) (3 points) Now, run experiments with the three different target distributions described above by changing the target distribution function. Include plots of the sampled distribution against the target distribution for different sample sizes (code for generating the plots for different number of samples is already included).

Images included in the github file.

c) (4 points) Interpret and discuss the results: in what scenarios was importance sampling the most/least effective?

Importance sampling was the most effective in the first scenario, the gaussian distribution. The weighted samples matched very closely with the target distribution because the proposal samples and target distribution had significant overlap, which in effect made the first scenario most effective. The least effect scenario was the third one that was a mixture of uniform distributions. The proposal samples did not overlap much with the target distribution so the weights were higher in variance. The second scenario, mixture of gaussians, was in the middle of both for effectiveness. The proposal and the target as some overlap, but it was at the 2 ends of the proposal, so the weights were skewed.

d) (5 points) Add code to compute estimates of the mean of each target distribution using Importance Sampling with 100000 samples. In which case are these estimates the most accurate?

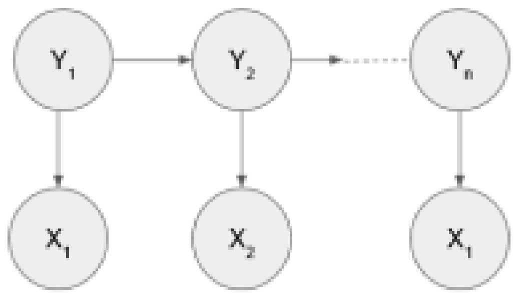
Code is on the github along with the visuals.

Similarly to (C), the estimates were the most accurate for target one which had the most overlap with the proposal and the target. The increase to 100,000 samples just reinforced the trend that occurred with 10,000 samples and increased the overlap with the target distribution, so the importance sampling was even more effective and accurate. The other two scenarios also benefited, but the relative accuracy between scenarios stayed the same.

e) (5 points) In which of the previous scenarios would Importance Sampling remain effective if the roles of the Gaussian proposal distribution and the target distributions were reversed? Consider the impact of distribution properties on the sampling efficiency and accuracy.

The scenario where importance sampling would still be effective is the first scenario. This is because even though the proposal and target are reversed they would both still be gaussian, so they would still overlap sufficiently for the weighted samples to be accurate/effective. For the other scenarios, since they are different distributions they would not overlap as much as the two gaussians, which will make importance less effective than the first scenario.

Question 4. Sunny, Rainy, or Ice-creamy? Modeling weather with HMMs (30 points) Johnathan, your friendly TA, is a fervent ice cream enthusiast. Johnathan indulges more as the mercury rises. Over n days, with a sequence of temperatures $Y_1, \ldots, Y_n$, Johnathan's corresponding ice cream consumption is $X_1 \ldots X_n$. Here, Johnathan's daily ice cream intake (X_t) can take values of 0, 1, or 2 servings, and the temperature for a day, Y_t can take values of 2 (cold), 1 (warm), or 0 (hot). This scenario is well-modeled by a *Hidden Markov Model* (HMM), where Y represents the *hidden states* (the temperature), influencing Johnathan's observable ice cream consumption (X). The temperature on any given day (Y_i) depends on the previous day's weather (Y_{i-1}), making $P(y_i|y_1, \ldots, y_{i-1}) = P(y_i|y_{i-1})$. Similarly, the number of ice creams Johnathan eats on day i, X_i, depends only on that day's temperature (Y_i), hence $P(x_i|y_1, \ldots, y_i) = P(x_i|y_i)$. An illustration of the HMM as a Bayesian network is shown.



The distribution $P(y_{i+1}|y_i)$ is called the *transition probability* and $P(x_i|y_i)$ are the *emission probability*. $P(y_0)$ will give the starting distribution. An HMM class is provided, with details of the transition probabilities ($P(y_{i+1}|y_i)$), emission probabilities ($P(x_i|y_i)$), and the initial state distribution ($P(y_0)$). For instance, transition probability[y1][y2] fetches the probability $P(Y_{i+1} = y_1|Y_i = y_2)$. The Google Colab link is: https://colab.research.google.com/drive/1MYiJENMv2l_lMVl9kuDzPmb0JyojcUdT.

Suppose we have observed the number of ice creams that Johnathan has eaten over n days $x_1 \ldots x_n$. Our aim is to deduce the most probable sequence of temperatures $y_1, \ldots, y_n$ on those days. In other words, we need to find $y_1 \ldots y_n$ that maximizes $P(y_1 \ldots y_n \mid x_1, \ldots x_n)$.

a) Model Understanding and Implementation

i) (3 points) For a chosen sequence of $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$, how would you compute the joint distribution $P(y_1, \ldots, y_n, x_1, \ldots, x_n)$?

To compute the joint distribution P(y_1,...,y_n,x_1,...,x_n) for a specific sequence of ice cream consumptions x_1,...,x_n and temperatures y_1,...,y_n in Jonathan's ice cream HMM, we apply the factorization P(y_1,...,y_n,x_1,...,x_n) = P(y_1) × P(x_1|y_1) × ∏_t=2^n [P(y_t|y_(t-1)) × P(x_t|y_t)]. This breaks down into: P(y_1) from the initial temperature, P(x_t|y_t) representing likelihood of Jonathan eating x_t servings given temperature y_t, and P(y_t|y_(t-1)) representing how likely temperature y_t follows y_(t-1). Multiplying these values in sequence gives the joint probability of that particular temperature sequence and ice cream consumption occurring together.

ii) (4 points) Implement the function joint prob to calculate $P(y_1, \ldots, y_n, x_1, \ldots, x_n)$. Inputs include outputs, a list of n observed ice cream consumptions, and hiddens, a list defining the hidden temperature states. Note: the parameters of functions are annotated with the expected types. Python will not check types, but the type annotations are guidelines.

iii) (5 points) Complete get likely hidden states, accepting an HMM object and observed con sumptions. For every possible temperature sequence (such as [0, 0, 0, 0, 0], [0, 0, 0, 0, 1] . . .), compute

$$P(y_1, \ldots, y_n, x_1, \ldots, x_n)$$

Then, determine

$$P(y_1, \ldots, y_n, x_1, \ldots, x_n)$$

$$P(y_1, \ldots, y_n|x_1, \ldots, x_n) = \frac{P(y_1, \ldots, y_n, x_1, \ldots, x_n)}{P(x_1, \ldots, x_n)},$$

You can find $P(x_1, \ldots x_n)$ by summing $P(y_1 \ldots y_n, x_1, \ldots x_n)$ over all possibilities of $y_1 \ldots y_n$. Sort by the conditional probability and output the 10 sequences of states $y_1, \ldots y_n$ which have the highest probabilities. Also output the corresponding joint probabilities. Note: Starter code has been provided. You cannot insert a list as a key in a Python dictionary, so you should convert it to a tuple first by using the tuple() function.

b) Gibbs Sampling

We will now use an alternative method to answer questions about the posterior distribution

($P(y_1, \ldots, y_n | x_1, \ldots, x_n$ using Gibbs sampling. Recall that we can use the algorithm to sample from a complex joint probability distribution.

i) (3 points). Within the inner loop of the Gibbs sampling algorithm you will need to sample a new y_i according to the distribution

$$P(Y_i = y \mid y_1 \ldots y_{i-1}, y_{i+1} \ldots y_n, x_1, \ldots x_n)$$

Explain why this is the same as $P(Y_i = y \mid Y_{i-1}, Y_{i+1}, X_i)$. (Why Y_i is independent on all other variables conditioned on Y_{i-1}, Y_{i+1}, X_i.)

Start by sampling each Y_i according to P(Y_i = y | y₁...y_(i-1), y_(i+1)...y_n, x_1...x_n). This distribution simplifies to P(Y_i = y | Y_(i-1), Y_(i+1), X_i) because of the conditional independence properties of HMMs. This occurs because once we condition on Y_(i-1) and Y_(i+1), the state Y_i is d-separated from all other variables in the model such that Y_i only directly depends on the previous temperature (Y_(i-1)), the next temperature (Y_(i+1)), and the ice cream consumption at time i (X_i). The Markov property ensures that Y_i is independent of all other temperatures given its immediate neighbors, while the emission model ensures that X_i depends only on Y_i.

ii) (5 points). Implement the function conditional weights, which takes the outputs and hidden state list and the required i. This must return the distribution $P(Y_i = y \mid Y_{i-1}, Y_{i+1}, X_i)$. This would be a list of 3 floats with the desired probabilities when $y = 0$, $y = 1$, $y = 2$ respectively.

You are given that

$$P(Y_i = y \mid Y_{i-1} = y_{i-1}, Y_{i+1} = y_{i+1}, X_i = x_i) \propto$$
$$P(Y_i = y \mid Y_{i-1} = y_{i-1}) P(Y_{i+1} = y_{i+1} \mid Y_i = y) P(X_i = x_i \mid Y = y)$$

If $i = 0$ then the first term will be $P(Y_0 = y)$. If $i = n - 1$ then the second term will be omitted. Note: You do not need to ensure the values sum to one.

iii) (5 points).

Implement the Gibbs sampling algorithm gibbs sampling, sampling $y_0, \ldots, y_{n-1}$ across iterations to estimate the most likely temperature sequences. To warm up, you can run the outer loop for the first 1000 or so iterations. Then for 5000 iterations, take a sample of $y_0 \ldots y_{n-1}$. Return a list of 5000 samples.

Note: you may use random.choices(population=population, weights=weights, k=1)[0] to sam ple a single value according to a probability weights.

iv) (5 points). Implement estimate likely hidden states. Run the gibbs sampling function and count the frequency of samples to estimate the probabilities $P(y_1 \ldots y_n \mid x_1 \ldots x_n)$ for each choice of $y_1 \ldots y_n$.

Sort by the joint probability and output the 10 sequence of states $y_1, \ldots y_n$ which have the highest probabilities. The top 10 states should be similar to the output of a)iii).