CS461 Homework 5

Mannan Shukla

December 16, 2024

Problem 1: Expectation-Maximization (EM) Algorithm

1.1 Log-Likelihood Computation

The log-likelihood for the initial parameters is given by:

$$\log L = \sum_{n=1}^{N} \log \left(\sum_{k=0}^{1} \pi_k \cdot \mathcal{N}(x_n | \mu_k, \sigma_k^2) \right)$$

where:

$$\mathcal{N}(x_n | \mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(x_n - \mu_k)^2}{2\sigma_k^2} \right)$$

Using the initial parameters:

$$\pi_0 = \pi_1 = 0.5, \quad \mu_0 = -1,\, \mu_1 = 1, \quad \sigma_0^2 = \sigma_1^2 = 1$$

and the data points $x = [2, 1, -1, -2]$, the initial log-likelihood is:

$$\log L = -7.158.$$

1.2 E-Step

The responsibilities γ_{nk} are computed as:

$$\gamma_{nk} = \frac{\pi_k \cdot \mathcal{N}(x_n | \mu_k, \sigma_k^2)}{\sum_{j=0}^{1} \pi_j \cdot \mathcal{N}(x_n | \mu_j, \sigma_j^2)}.$$

The resulting responsibilities are:

$$\gamma = \begin{bmatrix} 0.01799 & 0.98201 \\ 0.11920 & 0.88080 \\ 0.88080 & 0.11920 \\ 0.98201 & 0.01799 \end{bmatrix}.$$

1.3 M-Step

The updated parameters are computed as follows:

$$\pi_k(t+1) = \frac{1}{N}\sum_{n=1}^{N}\gamma_{nk},$$
$$\mu_k(t+1) = \frac{\sum_{n=1}^{N}\gamma_{nk}\cdot x_n}{\sum_{n=1}^{N}\gamma_{nk}},$$
$$\sigma_k^2(t+1) = \frac{\sum_{n=1}^{N}\gamma_{nk}\cdot (x_n-\mu_k(t+1))^2}{\sum_{n=1}^{N}\gamma_{nk}}.$$

The updated parameters are:

$$\pi_0 = \pi_1 = 0.5, \quad \mu_0 = -1.3448,\ \mu_1 = 1.3448, \quad \sigma_0^2 = \sigma_1^2 = 0.6914.$$

1.4 Log-Likelihood After Update

The updated log-likelihood is:

$$\log L = \sum_{n=1}^{N}\log\left(\sum_{k=0}^{1}\pi_k\cdot\mathcal{N}(x_n|\mu_k,\sigma_k^2)\right).$$

Substituting the updated parameters, we obtain:

$$\log L = -6.4619.$$

—

Problem 2: Exact vs. Approximate Inference

2.1 Variable Elimination

We compute:

$$P(\text{Cloudy} \mid \text{Sprinkler} = T, \text{WetGrass} = T).$$

Case 1: $C = T$

$$P(C=T,S=T,W=T,R=T) = 0.5\cdot 0.1\cdot 0.8\cdot 0.99 = 0.0396,$$
$$P(C=T,S=T,W=T,R=F) = 0.5\cdot 0.1\cdot 0.2\cdot 0.9 = 0.009.$$

$$P(C=T,S=T,W=T) = 0.0396 + 0.009 = 0.0486.$$

—

Case 2: $C = F$

$$P(C=F,S=T,W=T,R=T) = 0.5\cdot 0.5\cdot 0.2\cdot 0.99 = 0.0495,$$
$$P(C=F,S=T,W=T,R=F) = 0.5\cdot 0.5\cdot 0.8\cdot 0.9 = 0.18.$$

$$P(C=F,S=T,W=T) = 0.0495 + 0.18 = 0.2295.$$

Then Normalize,

$$P(C \mid S=T, W=T) = \frac{P(C, S=T, W=T)}{\sum_C P(C, S=T, W=T)}.$$

$$P(C=T \mid S=T, W=T) = \frac{0.0486}{0.0486 + 0.2295} \approx 0.1748, \quad P(C=F \mid S=T, W=T) = 1 - 0.1748 = 0.8252.$$

$$P(\text{Cloudy} = T \mid \text{Sprinkler} = T, \text{WetGrass} = T) \approx 0.1748,$$
$$P(\text{Cloudy} = F \mid \text{Sprinkler} = T, \text{WetGrass} = T) \approx 0.8252.$$

2.2 Gibbs Sampling

To approximate $P(\text{Cloudy} \mid \text{Sprinkler} = T, \text{WetGrass} = T)$, I used the following Python code for 1000 iterations:

The resulting posterior probabilities are:

$$P(\text{Cloudy} = T \mid \text{Sprinkler} = T, \text{WetGrass} = T) \approx 0.1778,$$
$$P(\text{Cloudy} = F \mid \text{Sprinkler} = T, \text{WetGrass} = T) \approx 0.8222.$$

Problem 3: VAE Evidence Lower Bound (ELBO)

Starting with the log marginal likelihood:

$\log p_\theta(x_i) = \log \sum_z p_\theta(x_i, z)$

Introduce the approximate posterior $q_\phi(z|x_i)$ and multiply/divide by it:

$$\log p_\theta(x_i) = \log \sum_z \frac{p_\theta(x_i, z) q_\phi(z|x_i)}{q_\phi(z|x_i)}.$$

By Jensen's inequality (concavity of the logarithm):

$$\log p_\theta(x_i) \geq \mathbb{E}_{q_\phi(z|x_i)} \left[\log p_\theta(x_i, z) - \log q_\phi(z|x_i)\right].$$

Expand the joint distribution $p_\theta(x_i, z) = p_\theta(z) p_\theta(x_i|z)$:

$$\log p_\theta(x_i) \geq \mathbb{E}_{q_\phi(z|x_i)} \left[\log p_\theta(z) + \log p_\theta(x_i|z) - \log q_\phi(z|x_i)\right].$$

Rearranging terms:

$$\log p_\theta(x_i) \geq -D_{KL}\left(q_\phi(z|x_i) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x_i)} \left[\log p_\theta(x_i|z)\right].$$

Thus, the ELBO (Evidence Lower Bound) is derived:

$$\log p_\theta(x_i) \geq \mathcal{L} = -D_{KL}\left(q_\phi(z|x_i) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x_i)} \left[\log p_\theta(x_i|z)\right].$$

—

```python
import numpy as np

# Given CPTs
P_C = 0.5  # P(C = T)
P_S_given_C = {True: 0.1, False: 0.5}
P_R_given_C = {True: 0.8, False: 0.2}
P_W_given_S_R = {
    (True, True): 0.99,  # P(W = T | S = T, R = T)
    (True, False): 0.9,  # P(W = T | S = T, R = F)
    (False, True): 0.9,  # P(W = T | S = F, R = T)
    (False, False): 0.0  # P(W = T | S = F, R = F)
}

# Fixed evidence
S = True  # Sprinkler = T
W = True  # WetGrass = T

# Initialize variables
C = np.random.choice([True, False])  # Cloudy
R = np.random.choice([True, False])  # Rain

# Conditional Sampling Functions
def sample_C(R):
    """Sample C given R, S = T, W = T."""
    P_C_T = P_C * P_S_given_C[True] * (P_R_given_C[True] if R else (1 - P_R_given_C[True]))
    P_C_F = (1 - P_C) * P_S_given_C[False] * (P_R_given_C[False] if R else (1 - P_R_given_C[False]))
    return np.random.choice([True, False], p=[P_C_T / (P_C_T + P_C_F), P_C_F / (P_C_T + P_C_F)])

def sample_R(C):
    """Sample R given C, S = T, W = T."""
    P_R_T = P_R_given_C[C] * P_W_given_S_R[(S, True)]
    P_R_F = (1 - P_R_given_C[C]) * P_W_given_S_R[(S, False)]
    return np.random.choice([True, False], p=[P_R_T / (P_R_T + P_R_F), P_R_F / (P_R_T + P_R_F)])

# Gibbs Sampling
iterations = 10000
cloudy_samples = []

for _ in range(iterations):
    C = sample_C(R)  # Sample C given R, S = T, W = T
    R = sample_R(C)  # Sample R given C, S = T, W = T
    cloudy_samples.append(C)

# Compute approximate posterior
P_C_T_approx = sum(cloudy_samples) / len(cloudy_samples)
P_C_F_approx = 1 - P_C_T_approx

print("Gibbs Sampling Results:")
print(f"P(Cloudy = T | Sprinkler = T, WetGrass = T): {P_C_T_approx:.4f}")
print(f"P(Cloudy = F | Sprinkler = T, WetGrass = T): {P_C_F_approx:.4f}")
~
~
~
~
"gibbs.py" 50L, 1720B written
```

Figure 1: Gibbs Sampling Code Implementation

Problem 4: RBM Movie Recommendation System

4.1 Bipolar Coding Necessity

Using ± 1 instead of 0/1 makes the Restricted Boltzmann Machine (RBM) more symmetric. When the visible and hidden units are centered around zero, the energy function and probability calculations become more balanced and mathematically cleaner. This symmetry can lead to simpler derivations and sometimes improved training stability.

4.2 Conditional Probability Formulation

When switching from binary $\{0, 1\}$ to bipolar $\{-1, +1\}$ coding, each 0 is replaced with -1 and 1 remains $+1$. To ensure that a zero-valued input corresponds to a 50% probability for $+1$ and 50% for -1, we rescale the input to the logistic (sigmoid) function by a factor of 2. Thus, for hidden units:

$$P(h_j = +1 \mid v) = \sigma\big(2(W_{j,:}v + c_j)\big),$$

and similarly for visible units:

$$P(v_i = +1 \mid h) = \sigma\big(2(W_{i,:}h + b_i)\big),$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$.

4.3 Predicting Preference for $\mathbf{m_2}$

Given: The user liked m_1 ($m_1 = +1$) and disliked m_3 ($m_3 = -1$). We want to predict the preference for m_2.

Steps:

1. **Known Visible Units:** Set $v = (m_1, m_2, m_3) = (+1, ?, -1)$. We initially treat m_2 as unknown.

2. **Hidden Input:** Compute $W^T v + c$. Given the parameters, this yields:
$$W^T v + c = [-2,\ 4,\ -2].$$

3. **Hidden Probabilities:** For each hidden unit h_j:
$$P(h_j = +1 \mid v) = \sigma(2(W_{j,:}v + c_j)) \implies h\_prob \approx [0.1192,\ 0.9820,\ 0.1192].$$

4. **Hidden States:** Since $0.1192 < 0.5$ and $0.9820 > 0.5$:
$$h_1 = -1, \quad h_2 = +1, \quad h_3 = -1.$$

5. **Visible Reconstruction:** Using h, compute $Wh + b$:
$$Wh + b = [1,\ -2,\ -6].$$
Apply σ (with factor 2 for bipolar):
$$v\_prob = [\sigma(1), \sigma(-2), \sigma(-6)] \approx [0.7311,\ 0.1192,\ 0.0025].$$

6. **Prediction for m_2:**
$$P(m_2 = +1) \approx 0.1192.$$
Since $0.1192 < 0.5$, we predict:
$$m_2 = -1 \quad \text{(user is predicted to dislike } m_2\text{)}.$$