Homework 5 - Introduction to Machine Learning For Engineers

kipngeno koech - bkoech

April 18, 2025

1 Gaussian Mixture Models

Consider an exponential mixture model for a 1-D dataset $\{x_n\}$ with the density function

$$p(x) = \sum_{k=1}^{K} \omega_k \text{Exp}(x|\mu_k),$$

where K is the number of mixture components, μ_k is the rate parameter, and ω_k is the mixture weight corresponding to the k-th component. The exponential distribution is given by

$$\text{Exp}(x|\mu) = \mu \exp(-x\mu) \quad \text{for all } x \geq 0. \tag{1}$$

We would like to derive the model parameters (ω_k, μ_k) for all k using the EM algorithm. Consider the hidden labels $z_n \in \{1, \ldots, K\}$ and indicator variables r_{nk} that are 1 if $z_n = k$ and 0 otherwise. The complete log-likelihood (assuming base e for the log) is then written as

$$\sum_n \log p(x_n, z_n) = \sum_n \sum_{z_n=k} \left[\log p(z_n = k) + \log p(x_n|z_n = k)\right].$$

1. Write down and simplify the expression for the complete log-likelihood for the exponential mixture model described above. Plugging the definition of the exponential distribution here immediately gives

$$\sum_n \log p(x_n, z_n) = \sum_k \sum_n r_{nk} \left[\log \omega_k + \log \text{Exp}(x_n|\mu_k)\right] = \sum_k \sum_n r_{nk} \left[\log \omega_k + \log \mu_k - x_n\mu_k\right].$$

2. Solve the M step of the EM algorithm and find μ_k for $k = 1, \ldots, K$ that maximizes the complete log-likelihood. Taking the derivative of the log-likelihood with respect to μ_k and setting it to zero, we have:

$$\frac{1}{\mu_k} \sum_n r_{nk} - \sum_n r_{nk} x_n = 0. \tag{2}$$

$$\mu_k = \frac{\sum_n r_{nk}}{\sum_n r_{nk} x_n}. \tag{3}$$

3. Perform the E step of the EM algorithm and write the equation to update the soft labels $r_{nk} = P(z_n = k|x_n)$. Using Bayes' rule, we have:

$$r_{nk} = \frac{P(x_n, z_n = k)}{P(x_n)} = \frac{\omega_k \mu_k \exp(-x_n\mu_k)}{\sum_{k'} \omega_{k'} \mu_{k'} \exp(-x_n\mu_{k'})}.$$

2 Eigen Faces

Face recognition is an important task in computer vision and machine learning. In this question, you will implement a classical approach called Eigenfaces. You will use face images from the Yale Face Database B, which contains face images from 10 people under 64 lighting conditions. Please include your code in the final PDF you turn in for full credit.

1. **Dataset.** Download the data file `face_data.mat`. It contains three sets of variables:

   - `image`: each element is a face image (50×50 matrix). You can use `matplotlib.pyplot.imshow` to visualize the image. The data is stored in a cell array.
   - `personID`: each element is the ID of the person, which takes values from 1 to 10.
   - `subsetID`: each element is the ID of the subset which takes values from 1 to 5. Here the face images are divided into 5 subsets. Each subset contains face images from all people, but with different lighting conditions.

2. [**10 points**] Implement PCA. Fill in the function `pca_fun` in the `pca.py` file. The function takes the data matrix (each row being a sample) and target dimensionality d (lower than or equal to the original dimensionality) as the input, and outputs the selected eigenvectors.

3. [**25 points**] Compute Eigenfaces. Take each 50×50 training image and vectorize it into a 2500- dimensional vector. Use your PCA implementation from part (b) to perform PCA on all vectorized face images, and retain the first $d = 200$ eigenvectors. These eigenvectors are called eigenfaces (when displayed as images). Please display the top 5 eigenfaces (use `imshow`) in your report.

3 Thompson Sampling

Consider the Thompson Sampling (TS) algorithm, a Bayesian approach to the multi-armed bandit problem. Consider a Bernoulli bandit with n arms, where each arm i at time-step $1 \leq t \leq T$ has Bernoulli i.i.d. rewards $r_{i,t} \in \{0, 1\}$ with $\mathbb{E}[r_{i,t}] = \mu_i$.

The TS algorithm starts with a prior distribution of μ_i for each arm i using the $P_{i,0} \sim \text{Beta}(1, 1)$ distribution and proceeds by selecting an arm based on the posterior distribution as follows. Note the prior distribution of μ_i at time t is denoted as $P_{i,t-1}$ and the posterior as $P_{i,t}$. Further, the posterior of the current time-step becomes the prior for the next time-step.

Algorithm 1 Thompson Sampling

1: **for** $t = 1, 2, \ldots, T$ **do**
2: Sample $\hat{\mu}_{i,t} \sim P_{i,t-1}$ for each arm $i \in \{1, \ldots, n\}$
3: Play arm $i_t = \arg\max_i \hat{\mu}_{i,t}$
4: Observe reward $r_{i_t,t}$ and update posterior $P_{i,t}$
5: **end for**

Recall the probability density function of the Beta distribution, Beta(α, β), for any $x \in [0, 1]$ is

$$p(x) = \frac{(\alpha + \beta - 1)!}{(\alpha - 1)!(\beta - 1)!} x^{\alpha-1}(1 - x)^{\beta-1}.$$

We also know, for any $p, q \geq 1$,

$$\int_0^1 x^p (1 - x)^q \, dx = \frac{(p + q + 1)!}{(p - 1)!(q - 1)!}.$$

1. **[15 points]** Until time-step t, suppose arm $i \in \{1, \ldots, n\}$ is pulled $N_{i,t}$ times and its total observed reward is

$$S_{i,t} := \sum_{u \leq t : i_u = i} r_{i,u},$$

where i_u represents the arm chosen at time-step u. Find $P_{i,t}$, the posterior distribution of μ_i, given the Beta prior as described above and observations on the rewards until time-step t. (Hint: Compute the posterior for the first time-step. Use this recursively for the following time-steps.)

2. **[5 points]** Compute the mean and variance of the posterior distribution of μ_i found in part (a).

3. **[5 points]** Using the computations in part (b), explain how TS balances exploration and exploitation.