Exponential Mixture Models: Derivations, Coding, and Data Analysis

STA 325

The goal of this exercise will be to work through how derivations connect with coding up functions and then revisiting a case study we have looked at before. This is how machine learning often works in practical settings, and the goal is to connect all aspects of learning.

Model Setup

Consider a data set $\{x_1, x_2, \ldots, x_n\}$ generated from a mixture of K exponential distributions. The mixture model is given by:

$$p(x|\theta) = \sum_{k=1}^{K} \pi_k \lambda_k e^{-\lambda_k x}$$

where:

- π_k is the mixing coefficient for the k-th component (with $\sum_{k=1}^{K} \pi_k = 1$),
- λ_k is the rate parameter of the k-th exponential distribution,
- $\theta = \{\pi_k, \lambda_k\}$ are the parameters we want to estimate.

Part I

1. Write the mixture model using an unknown (or latent) variable/component or two-stage approach so that we can estimate any unknown parameters. Assume that Z_i indicates the unknown (or latent) component that x_i is drawn from.
2. Write the marginal distribution of $P(x_i)$.
3. Derive the expectation (E-step) and maximization (M-step) steps of the EM algorithm.

a. Specifically for the E-step, using Bayes' Theorem, show that

$$\gamma_{ik} = P(Z_i = k|x_i) = \frac{\pi_k \lambda_k e^{-\lambda_k x_i}}{\sum_{j=1}^{K} \pi_j \lambda_j e^{-\lambda_j x_i}}.$$

b. Show that the M-step is as follows (given the E-step):

The mixing coefficient for each component is updated as follows:

$$\pi_k = \frac{1}{n} \sum_{i=1}^{n} \gamma_{ik}.$$

Show that the rate parameters are updated as follows:

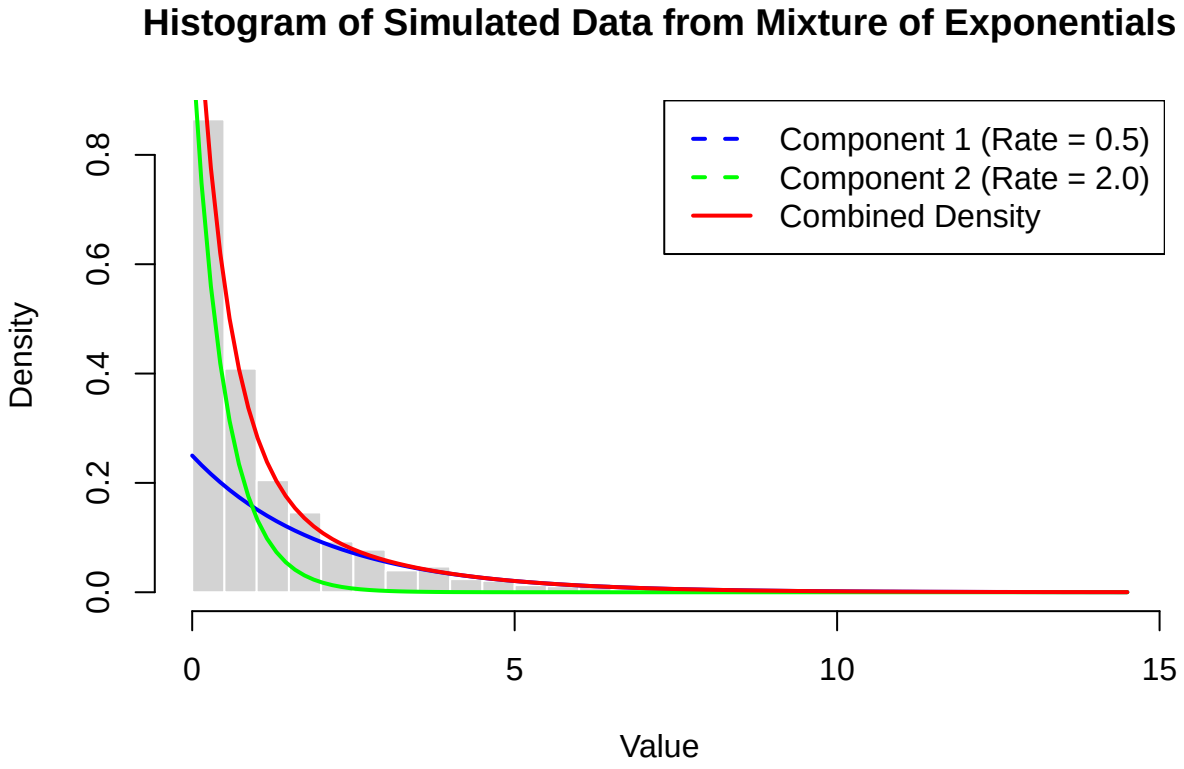
$$\lambda_k = \frac{\sum_{i=1}^{n} \gamma_{ik}}{\sum_{i=1}^{n} \gamma_{ik} x_i}.$$

4. Write the log-likelihood, which is used to to evaluate convergence of the EM algorithm. Explain how convergence can be checked using the log-likelihood.

Part II

Implement a two-component exponential mixture model (using Part I), estimating the unknown parameters using the EM algorithm. You will need to write your own code to do this.

Test your code on a simulated data from an exponential mixture model, where we know what the estimated parameters should be close to, assuming the algorithm does not get stuck in a local mode. You will find such an example below.



Hint: You could utilize the skeleton code below if you find it helpful or start with your own.

```r
# Function to fit a k-component exponential mixture model
mixtureExponetials <- function(data, K, max_iter = 1000, tol = 1e-5) {
  n <- length(data)
  pi <- rep(1/K, K)  # Mixing proportions
  lambda <- runif(K, 0.1, 1)  # Rate parameters

  log_likelihoods <- numeric(max_iter)  # To store log-likelihood values

  # E-step: find the mixing proportions and normalize these
  # (this is the result from Bayes Theorem in part I)

  # M-step: update the mixing proportions, update the rate parameters
  # (These are from the M-step in part I)


  # Calculate the log-likelihood for the current iteration

  # Check for convergence. This point can be a bit tricky
  # and you may run into errors here potentially.

  return(list(pi = pi, lambda = lambda, log_likelihood = log_likelihoods))
}
```

Remark: I often find it helpful to write and de-bug code in base R (instead of RStudio personally), however, this is all personal preference. Why?

1. I find it personally easier to debug and go through.
2. I find the debugging functions a bit easier to understand when I have an error, but this is all personal preference.
3. Find what works the best for you as it's all about finding the best workflow, and it's different for everyone.
4. If you have questions about workflow, please do ask. I love talking about it and it's a very important part of machine learning.

Remark: I often find markdown/quarto helpful for visualization or tidying up data to showcase, whereas, I find base R most effective for writing code for papers/projects. If you look into how packages are written in R, they are in base R, however vignettes are showcased in markdown. Think of why this would be the case.

Part III

Now, consider a second simulation study. Test the EM algorithm out on this simulated data set and report your findings.

a. Report the parameter estimates for five different starting values of the EM algorithm. Explain your findings and behavior. It may be helpful to look at the log-likelihood plot.

b. You should find in part a) that your estimates are in accurate but do your best to try and explain why they are not accurate. Explain what is happening to the EM algorithm that causes the estimates to be inaccurate.

c. Do you have a suggestion for improving your estimates? Hint: Think about k-means. Make a simple modification to your code. Do you find improvements? Or do you still have similar problems that you ran into in part b.

d. Explain what this has taught you about mixture models and the EM algorithm from the two different simulation studies. Given this new knowledge, what precautions should you take when working with mixture models and the EM algorithm moving forward?

```r
# Simulation parameters
set.seed(123)  # For reproducibility
n_samples <- 1000  # Number of samples
true_rates <- c(0.5, 1.5)  # True rate parameters for two components
true_proportions <- c(0.6, 0.4)  # True mixing proportions

# Generate synthetic data from a mixture of two exponential distributions
data <- c(rexp(n_samples * true_proportions[1], rate = true_rates[1]),
          rexp(n_samples * true_proportions[2], rate = true_rates[2]))
```