Statistical Computing Project 4

Mario Ibanez

March 26, 2016

Introduction

The purpose of this report is to evaluate the effectiveness of the EM algorithm in solving a mixture model problem. The problem that will be solved in this report is that given a data set, attempt to estimate the proportion at which data was sampled from two normal distributions along with estimating the means and standard deviations of those two distributions. More explicitly, the data will be sampled from the distribution:

$$f(x) = w_1 N(\mu_1, \sigma_1) + w_2 N(\mu_2, \sigma_2) = w_1 f_1(x) + w_2 f_2(x)$$

and the EM algorithm will be used to estimate the parameters w_1, w_2, μ_1, μ_2, σ_1 and σ_2. In addition to estimating these parameters, an attempt can be made at evaluating the algorithm itself by calculating various statistics reflecting the algorithm's performance.

Methods

In order to evaluate the algorithm, known values for the parameters listed above will be used to generate data. The algorithm will then use this data to estimate the six unknown mixture model parameters. However, because of the constraint that $w_1 + w_2 = 1$, there are in fact only five unknown parameters. If the true values for these parameters are known, the algorithm can be evaluated for effectiveness by comparing the estimates it generates to the true values. A method that will be used to evaluate the accuracy of the algorithm will be "z-scores" for the estimated means. Explicitly:

$$z_1 = \frac{\hat{\mu_1} - \mu_1}{\sigma_1} \text{ and } z_2 = \frac{\hat{\mu_2} - \mu_2}{\sigma_2}$$

These z-scores will measure how far away from the true values the estimates of the means were.

Though there are different approaches possible in order to evaluate the EM algorithm's ability to solve a mixture model problem, the one used in this report is to run the algorithm repeatedly for different samples from the same distribution. This will give an idea, for that specific distribution, how the algorithm tends to perform. Another alternative approach that will not be taken in this report is to run the algorithm a single time for different distributions and compare the algorithm's ability to solve the problem depending on the parameters.

Data

The data used in this report to evaluate the EM algorithm will be 100 values sampled from the following mixture distribution:

$$f(x) = (0.30)N(-1, 2) + (0.70)N(2, 1)$$

In order to perform this sampling, a random number will be sampled from the binomial distribution with parameters $n = 100$ and $p = 0.30$. This value will be used to determine how many values are sampled from $N(-1, 2)$. The rest of the values will then be sampled from $N(2, 1)$. Sampling from a normal distribution with any parameters is easy in the *R* language using a built in function. The code below generates the data:

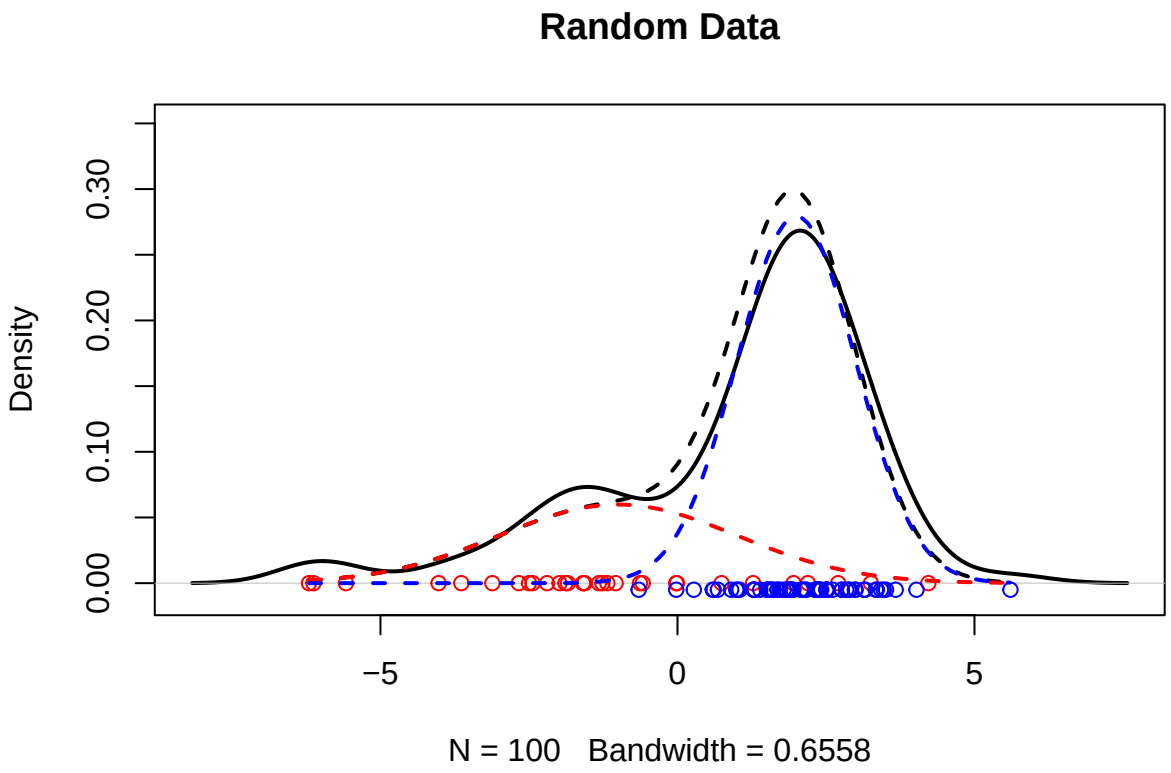
```r
# Mixture weights and total sample size
proportion <- 0.30
sample_size <- 100

# Means and standard deviations
mu_1 <- -1; mu_2 <- 2
sigma_1 <- 2; sigma_2 <- 1

# Calculate how many values will come from each component
sample1_size <- rbinom(n = 1, size = sample_size, prob = proportion)
sample2_size <- sample_size - sample1_size

# Sample that many values from each respective normal distribution
sample1 <- rnorm(n = sample1_size, mean = mu_1, sd = sigma_1)
sample2 <- rnorm(n = sample2_size, mean = mu_2, sd = sigma_2)
```

The plot below allows us to visualize the data set that will be used later in the report to evaluate the EM algorithm:



In the plot, the dotted lines represent the true population distributions. The red dotted line is the PDF of $N(-1, 2)$ while the blue dotted line is the PDF of $N(2, 1)$ and the black dotted line is the PDF of $(0.30)N(-1, 2) + (0.70)N(2, 1)$. The red and blue points at the bottom of the plot represent the generated data. Finally, the black solid line is a plot of the density of the generated data. Notice of course that while the black dotted line and the black solid line should be similar in shape, there is no guarantee that from sample to sample this will occur. This is especially true when the two distributions significantly overlap as they do in the example in this report.

EM Algorithm

The EM algorithm works by alternating between two steps until convergence. The first step is the Expectation step and the second step is the Maximization step. In this applicaton of the algorithm, the expectation step involves the relative probability that a data point was generated from the first distribution. (Note, the first distribution is $f_1(x) = N(-1, 2)$) This value is calcuated for each data point in each iteration. The formula for the i^{th} value is:

$$\hat{z_i} = \frac{\hat{w_1} f_1(x_i)}{\hat{w_1} f_1(x_i) + (1 - \hat{w_1}) f_2(x_i)}$$

This value is used to weight the estimates for the parameters μ_1, μ_2, σ_1, and σ_2. The next step is the maximization step. Updated estimates of the five parameters are calculated:

$$\hat{\mu_1} = \frac{\sum \hat{z_i} x_i}{\sum \hat{z_i}} \text{ and } \hat{\mu_2} = \frac{\sum (1 - \hat{z_i}) x_i}{\sum (1 - \hat{z_i})}$$

$$\hat{\sigma_1^2} = \frac{\sum \hat{z_i} (x_i - \hat{\mu_1})^2}{\sum \hat{z_i}} \text{ and } \hat{\sigma_2^2} = \frac{\sum (1 - \hat{z_i}) (x_i - \hat{\mu_1})^2}{\sum (1 - \hat{z_i})}$$

$$\hat{w_1} = \frac{\sum \hat{z_i}}{N} \text{ where } N = 100$$

Written more explictily, the steps are:
1) Initialize the five unknown paramater values
2) Calculate $\hat{z_i}$ for each value
3) Calculate new esimates for the five parameters
4) Return to step 2 if convergence criteria has not been met

The algorithm will run until the difference between to successive estimates of μ_1 differ by less than 0.01. This is an arbitrary convergence criteria but will serve fine for the purpose of this report.

Simulation & Results

As was partly mentioned earlier in the paper, the EM algorithm will be evaluated by repeatedly generating new random data from the distribution:

$$f(x) = (0.30)N(-1, 2) + (0.70)N(2, 1)$$

The "z-scores" defined earlier will be calculated each time. Random data will be generated 500 times and the algorithm allowed to run until reaching its convergence criteria each time. The initial values for the parameters will be the same in each trial: $\mu_1 = -0.5$, $\mu_2 = 0.5$, $\sigma_1 = 1$, $\sigma_2 = 1$ and $w_1 = 0.5$. The simulations will be done in R by using a loop to repeatedly generate random data, rerun the algorithm, and store the z scores for that trial. The code below accomplishes this:

```
z_scores <- data.frame(z1 = rep(0, 500),
                       z2 = rep(0, 500))

for(i in 1:500){
  # Initial parameter values guesses c(mu_1, mu_2, sigma_1, sigma_2, w1)
  parameters <- c(-0.5, 0.5, 1, 1, 0.5)
```

```
  # Calculate how many values will come from each component
  sample1_size <- rbinom(n = 1, size = sample_size, prob = proportion)
  sample2_size <- sample_size - sample1_size

  # Sample that many values from each respective normal distribution
  sample1 <- rnorm(n = sample1_size, mean = mu_1, sd = sigma_1)
  sample2 <- rnorm(n = sample2_size, mean = mu_2, sd = sigma_2)

  #while(TRUE){}
}
```

Conclusion

Plotting the true distribution