EM

Yaqiong Yao

10/12/2018

EM Algorithm for finite mixture regression

Validating E and M Step

E-step :

$$
\begin{aligned}
Q(\Psi|\Psi^{(k)}) &= \mathbb{E}[\log L(\Psi|\mathbf{X}, \mathbf{y}, \mathbf{Z})|\mathbf{X}, \mathbf{y}, \Psi] \\
&= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \mathbf{y}, \Psi^{(k)}) \log \prod_{i=1}^{n} p(\mathbf{z_i}, y_i|\mathbf{x_i}, \Psi) \\
&= \sum_{i=1}^{n} \sum_{\mathbf{z_i}} p(\mathbf{z_i}|\mathbf{x_i}, y_i, \Psi^{(k)}) \log p(\mathbf{z_i}, y_i|\mathbf{x_i}, \Psi)
\end{aligned}
$$

Here,

$$
\begin{aligned}
p(\mathbf{z_i} = \mathbf{z}, y_i|\mathbf{x_i}, \Psi^{(k)}) &= p(z_{ij} = 1, y_i|\mathbf{x_i}, \Psi^{(k)}) \\
&= p(z_{ij} = 1|\Psi^{(k)}) p(y_i|\mathbf{x_i}, \Psi^{(k)}, z_{ij} = 1) \\
&= \pi_j^{(k)} \phi(y_i - \mathbf{x_i}^T \beta_j^{(k)}; 0, \sigma^{2(k)}),
\end{aligned}
$$

where $\mathbf{z}$ is one choice of $\mathbf{z_i}$ and jth element is 1.

$$
\begin{aligned}
p(\mathbf{z_i} = \mathbf{z}|\mathbf{x_i}, y_i, \Psi^{(k)}) &= p(z_{ij} = 1|\mathbf{x_i}, y_i, \Psi^{(k)}) \\
&= \frac{p(z_{ij} = 1, y_i|\mathbf{x_i}, \Psi^{(k)})}{\sum_{z_i} p(\mathbf{z_i}, y_i|\mathbf{x_i}, \Psi^{(k)})} \\
&= \frac{\pi_j^{(k)} \phi(y_i - \mathbf{x_i}^T \beta_j^{(k)}; 0, \sigma^{2(k)})}{\sum_{j=1}^{m} \pi_j^{(k)} \phi(y_i - \mathbf{x_i}^T \beta_j^{(k)}; 0, \sigma^{2(k)})}
\end{aligned}
$$

Overall,

$$
Q(\Psi|\Psi^{(k)}) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}^{(k)} \{\log \pi_j + \log \phi(y_i - \mathbf{x_i}^T \beta_j; 0, \sigma^2)\}
$$

$$
p_{ij}^{(k+1)} = \frac{\pi_j^{(k)} \phi(y_i - \mathbf{x_i}^T \beta_j^{(k)}; 0, \sigma^{2(k)})}{\sum_{j=1}^{m} \pi_j^{(k)} \phi(y_i - \mathbf{x_i}^T \beta_j^{(k)}; 0, \sigma^{2(k)})}
$$

M-step :

$$
\begin{aligned}
Q(\Psi|\Psi^{(k)}) &= \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)}\{\log \pi_j + \log \phi(y_i - \mathbf{x_i}^T\beta_j; 0, \sigma^2)\} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} \log \pi_j - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} \log(2\pi\sigma^2) - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} \frac{(y_i - \mathbf{x_i}^T\beta_j)^2}{\sigma^2} \\
&= I_1 - \frac{1}{2}I_2 - \frac{1}{2}I_3
\end{aligned}
$$

For $\pi_j^{(k+1)}$, only I_1 contains it. Note that $\sum_{j=1}^{m}\pi_j = 1$. Thus the maximization can be found by finding solution for

$$
\frac{\partial \mathcal{L}(\pi_1, ..., \pi_m)}{\partial \pi_i} = 0,
$$

where

$$
\mathcal{L}(\pi_1, ..., \pi_m) = \sum_{i=1}^{n} p_{ij}^{(k)} \log \pi_j - \lambda \left\{ \sum_{j=1}^{m} \pi_j - 1 \right\}
$$

with λ a lagrange mulitplier. Thus its minimizier is

$$
\pi_j^{(k+1)} = \frac{\sum_{i=1}^{n} p_{ij}^{(k)}}{n}
$$

For $\beta_j^{(k+1)}$, only I_3 contains β_j. We can treat it as a weighted least square regression. Then

$$
\beta_j^{(k+1)} = (\sum_{i=1}^{n} x_i x_i^T p_{ij}^{(k)})^{-1} (\sum_{i=1}^{n} x_i p_{ij}^{(k)} y_i)
$$

For ${\sigma^2}^{(k+1)}$, we choose to minimize $-I_2 - I_3$. Take derivative with respect to σ^2, then

$$
\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)}}{\sigma^2} - \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} (y_i - \mathbf{x_i}^T\beta_j)^2}{\sigma^4} = 0
$$

$$
\sigma^2 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} (y_i - \mathbf{x_i}^T\beta_j)^2}{n}
$$

Thus,

$$
{\sigma^2}^{(k+1)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k)} (y_i - \mathbf{x_i}^T\beta_j^{(k)})^2}{n}
$$

Implement into function

```
regmix_em <- function(y, xmat, pi.init, beta.init, sigma.init, control){
  x <- as.matrix(xmat)
  n <- nrow(x)
  m <- length(pi)
  d <- ncol(x)
  itr <- 0
  pi <- as.vector(pi.init)
  beta <- beta.init
  sigma2 <- sigma.init^2
  diff <- 1

  while (diff >= control$tol) {
    piji <- matrix(rep(pi, n), ncol = m,  byrow = T) * (1/sqrt(2*(base::pi)*sigma2^2))*exp(-(y- x%*%beta
    pij <- piji/rowSums(piji)

    pi_new <- colMeans(pij)
    beta_new <- matrix(0, nrow = 2, ncol = 3)
    for (j in 1:m) {
      beta_new[,j] <- solve(t(x) %*% diag(pij[,j]) %*% x) %*% t(x) %*% diag(pij[,j]) %*% y
    }
    sigma2_new <- sqrt(sum(pij * ((y- x%*%beta)^2))/n)
    diff <- sum(abs(pi_new-pi)) + sum(abs(beta_new-beta)) + sum(abs(sigma2_new-sigma2))
    itr <- itr + 1
    if(itr >= control$maxit) break

    sigma2 <- sigma2_new
    pi <- pi_new
    beta <- beta_new
  }

  return(list(pi = pi, sigma2 = sigma2, beta = beta, itr = itr))
}
```

Generate data from mixture regression model

```
regmix_sim <- function(n, pi, beta, sigma) {
    K <- ncol(beta)
    p <- NROW(beta)
    xmat <- matrix(rnorm(n * p), n, p) # normal covaraites
    error <- matrix(rnorm(n * K, sd = sigma), n, K)
    ymat <- xmat %*% beta + error # n by K matrix
    ind <- t(rmultinom(n, size = 1, prob = pi))
    y <- rowSums(ymat * ind)
    data.frame(y, xmat)
}
```

```
n <- 400
pi <- c(.3, .4, .3)
bet <- matrix(c( 1,  1,  1,
                -1, -1, -1), 2, 3)
sig <- 1
set.seed(1205)
```

```r
dat <- regmix_sim(n, pi, bet, sig)

regmix_em(y = dat[,1], xmat = dat[,-1],
          pi.init = pi / pi / length(pi),
          beta.init = matrix(c( 1,  1,  1, -1, -1, -1), 2, 3),
          sigma.init = sig / sig,
          control = list(maxit = 500, tol = 1e-5))
```

```
## $pi
## [1] 0.3858218 0.2687874 0.3453908
##
## $sigma2
## [1] 1.023598
##
## $beta
##           [,1]       [,2]       [,3]
## [1,] 0.8796608  0.9911851 -0.9136977
## [2,] 0.9341964 -1.2424569 -1.1990372
##
## $itr
## [1] 49
```