Homework4

Hukai Luo

12 October 2018

1 Question

Given n independent observations of the response $Y \in \mathbb{R}$ and predictor $\mathbf{X} \in \mathbb{R}^p$, multiple linear regression models are commonly used to explore the conditional mean structure of Y given $\mathbf{X}$. However, in many applications, the underlying assumption that the regression relationship is homogeneous across all the observations $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$ can be easily violated. Instead, the observations may form several distinct clusters indicating mixed relationships between the response and the predictors. Such heterogeneity can be more appropriately modeled by a **finite mixture regression model**, consisting of, say, m homogeneous groups/components.

Suppose the density of y_i (conditional on $\mathbf{x}_i$), is given by

$$f(y_i \mid \mathbf{x}_i, \boldsymbol{\Psi}) = \sum_{j=1}^{m} \pi_j \phi(y_i; \mathbf{x}_i^\top \boldsymbol{\beta}_j, \sigma^2), \qquad i = 1, \ldots, n, (\#eq:mixregequal) \tag{1}$$

where π_js are called mixing proportions, $\boldsymbol{\beta}_j$ is the regression coefficient vector for the jth group, $\phi(\cdot\,; \mu, \sigma^2)$ denotes the density function of $N(\mu, \sigma^2)$, and $\boldsymbol{\Psi} = (\pi_1, \boldsymbol{\beta}_1, \ldots, \pi_m, \boldsymbol{\beta}_m, \sigma)^T$ collects all the unknown parameters.

1. Follow the lecture notes to verify the validity of the provided E- and M-steps. That is, derive the updating rules in the given algorithm based on the construction of an EM algorithm.
2. Implement this algorithm in R with a function `regmix_em`. The inputs of the functions are `y` for the response vector, `xmat` for the design matrix, `pi.init` for initial values of π_j's (a vector of $K \times 1$ vector), `beta.init` for initial values of $\boldsymbol{\beta}_j$'s (a matrix of $p \times K$ where p is `ncol(xmat)` and K is the number of components in the mixture), `sigma.init` for initial values of σ, and a `control` list for controlling max iteration number and convergence tolerance. The output of this function is the EM estimate of all the parameters.
3. Here is a function to generate data from the 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)
}
```

Generate data with the following and estimate the parameters.

```r
n <- 400
pi <- c(.3, .4, .3)
bet <- matrix(c( 1,  1,  1,
                -1, -1, -1), 2, 3)
sig <- 1
set.seed(1205)
dat <- regmix_sim(n, pi, bet, sig)
## regmix_em(y = dat[,1], xmat = dat[,-1],
##           pi.init = pi / pi / length(pi),
##           beta.init = bet * 0,
##           sigma.init = sig / sig,
##           control = list(maxit = 500, tol = 1e-5))
```

2 Verify the validity of the provided E- and M-steps

Based on the question, we have these formulas:
he density of y_i (conditional on $\mathbf{x}_i$)

$$f(y_i|x_i,\Psi) = \sum_{j=1}^{m} \pi_j \phi(y_i; x_i^T\beta_j, \sigma^2)$$

The complete log-likelihood can be written as

$$l_c^n(\Psi) = \sum_{i=1}^{n}\sum_{j=1}^{m} z_{ij} \ln\{\pi_j \phi(y_i - x_i^T\beta_j; 0, \sigma^2)\}$$

In the E-step, we calculate the conditional expectation of the complete log-likelihood

$$\begin{aligned}
E(z_{ij}|(x,y),\Psi^{(k)}) &= P(z_{ij}=1|(x_i,y_i),\Psi^{(k)}) \\
&= \frac{P(x_i,y_i,z_{ij}=1|\Psi^{(k)})}{P(x_i,y_i|\Psi^{(k)})} \\
&= \frac{\pi_j^{(k)}\varphi(y_i - x_i^T\beta_j; 0,\sigma^2)}{\sum_{j=1}^{m}\pi_j^{(k)}\varphi(y_i - x_i^T\beta_j; 0,\sigma^2)} \\
&= p_{ij}^{(k+1)}
\end{aligned}$$

So we can verify the

$$\begin{aligned}
Q(\Psi|\Psi^{(k)}) &= E[l_n^c(\Psi)|(x,y)] \\
&= \sum_z P(z|(x,y),\Psi^{(k)}) l_n^c(\Psi) \\
&= \sum_z P(z|(x,y),\Psi^{(k)}) \sum_{i=1}^{n}\sum_{j=1}^{m} z_{ij}\{log\pi_j + log\varphi(y_i - x_i^T\beta_j; 0,\sigma^2)\} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m}[\sum_z z_{ij}P(z|(x,y),\Psi^{(k)})]\{log\pi_j + log\varphi(y_i - x_i^T\beta_j; 0,\sigma^2)\} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m} E(z_{ij}|(x,y),\Psi^{(k)})\{log\pi_j + log\varphi(y_i - x_i^T\beta_j; 0,\sigma^2)\}
\end{aligned}$$

In the M-step, we will calculate $\pi_j^{(k+1)}$:

$$
\begin{aligned}
\frac{\partial Q}{\partial \pi_j} &= \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} \ln \pi_j \\
&= \sum_{j=1}^{m}(\ln \pi_j \sum_{i=1}^{n} p_{ij}^{(k+1)}) \\
&= \ln \pi_1 \sum_{i=1}^{n} p_{i1}^{(k+1)} + ... + \ln \pi_j \sum_{i=1}^{n} p_{ij}^{(k+1)} + ... \ln \pi_m \sum_{i=1}^{n} p_{im}^{(k+1)} \\
&= \ln \pi_1 \sum_{i=1}^{n} p_{i1}^{(k+1)} + ... + \ln \pi_j \sum_{i=1}^{n} p_{ij}^{(k+1)} + ... \ln(1 - \pi_1 - ... - \pi_j - ... - \pi_{m-1}) \sum_{i=1}^{n} p_{im}^{(k+1)} \\
&= 0
\end{aligned}
$$

$$
\pi_m^{(k+1)} = 1 - \sum_{j=1}^{m-1} \pi_j^{(k+1)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} - \sum_{j=1}^{m-1}\sum_{i=1}^{n} p_{ij}^{(k+1)}}{n} = \frac{\sum_{i=1}^{n} p_{im}^{(k+1)}}{n}
$$

and $\beta_j^{(k+1)}$:

$$
\begin{aligned}
\frac{\partial}{\partial \beta_j} Q(\Psi|\Psi^{(k)}) &= \frac{\partial I_3}{\partial \beta_j} \\
&= -2 \sum_{i=1}^{n} p_{ij}^{(k+1)} x_i (y_i - x_i^T \beta_j) \\
&= 0
\end{aligned}
$$

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

finally, the ${\sigma^2}^{(k+1)}$:

$$
\begin{aligned}
&\frac{\partial Q}{\partial \sigma^2} = -\frac{1}{2\sigma^2} \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} + \frac{1}{2(\sigma^2)^2} \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} (y_i - x_i^T \beta_j^{(k+1)})^2 = 0 \\
\Rightarrow & \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} (y_i - x_i^T \beta_j^{(k+1)})^2}{\sigma^2} = n \\
\Rightarrow & {\sigma^2}^{(k+1)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}^{(k+1)} (y_i - x_i^T \beta_j^{(k+1)})^2}{n}
\end{aligned}
$$

3 Implement this algorithm in R with a function `regmix_em`

Function factors:
`y` for the response vector, `xmat` for the design matrix, `pi.init` for initial values of π_j's (a vector of $K \times 1$ vector), `beta.init` for initial values of $\boldsymbol{\beta}_j$'s (a matrix of $p \times K$ where p is `ncol(xmat)` and K is the number of components in the mixture), `sigma.init` for initial values of σ, and a `control` list for controlling max iteration number and convergence tolerance.

```
regmix_em=function(y,xmat,pi.init,beta.init,sigma.init,control=list(maxit=500,tol=1e-5)){
  maxit=control$maxit
  tol=control$tol
  n=nrow(xmat)
  p=ncol(xmat)
  m=length(pi.init)
  x=as.matrix(xmat)
  pi=pi.init
  beta=beta.init
  sigma=sigma.init

  P=matrix(0,n,m)
  beta.new=matrix(0,p,m)
  count=0

  for(k in 1:maxit){
    for(i in 1:n){
      P[i, ]=pi*dnorm(y[i]-x[i, ]%*%beta,0,sigma)/sum(pi*dnorm(y[i]-x[i, ]%*%beta,0,sigma))
    }

    pi.new=colMeans(P)

    for(j in 1:m){
      beta.new[ ,j]<-solve(t(x)%*%diag(P[,j])%*%x)%*%t(x)%*%diag(P[,j])%*%y
    }

    sigma.new=sqrt(sum((P*(y%*%t(rep(1,m))-x%*%beta.new)^2))/n)

    if(sum(abs(pi.new-pi))+sum(abs(beta.new-beta))+sum(abs(sigma.new-sigma))<tol)break
      pi=pi.new
    beta=beta.new
    sigma=sigma.new
  }
  if(k==maxit)
  print("reach maximun loop")
  list(pi=pi.new,beta=beta.new,sigma=sigma.new,iteration=k)
}
```

4 Generate data with the following and estimate the parameters

Generate data with the following code:

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

Estimate the parameters using the sample function below:

```
regmix_sim <- function(n, pi, beta, sigma) {
  K <- ncol(beta)
  p <- nrow(beta)
  xmat <- matrix(rnorm(n * p), n, p)
  error <- matrix(rnorm(n * K, sd = sigma), n, K)
  ymat <- xmat %*% beta + error
  ind <- t(rmultinom(n, size = 1, prob = pi))
  y <- rowSums(ymat * ind)
  data.frame(y, xmat)
}
regmix_em(y = dat[,1], xmat = dat[,-1],
           pi.init = pi / pi / length(pi),
           beta.init = bet*0,
           sigma.init = sig / sig,
           control = list(maxit = 500, tol = 1e-5))
```

```
## $pi
## [1] 0.3333333 0.3333333 0.3333333
##
## $beta
##            [,1]       [,2]       [,3]
## [1,]  0.3335660  0.3335660  0.3335660
## [2,] -0.4754645 -0.4754645 -0.4754645
##
## $sigma
## [1] 1.732492
##
## $iteration
## [1] 2
```

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

```
## $pi
## [1] 0.3858218 0.2687873 0.3453909
```

```
##
## $beta
##           [,1]       [,2]       [,3]
## [1,] 0.8796608  0.9911852 -0.9136977
## [2,] 0.9341964 -1.2424569 -1.1990372
##
## $sigma
## [1] 1.023598
##
## $iteration
## [1] 49
```