Poisson Mixture EM Estimation

Lei Kang

Friday 9th December, 2016

(1)
Suppose we can observe independent X_i, but cannot observe the hidden variable Y. Define a complete data as $Z = (X, Y)$. Here $Y_i = 0$ if this observation comes from Poisson(μ_0) and $Y_i = 1$ if this observation comes from Poisson(μ_1). And in this case, since we are dealing with a mixture distribution of two Possions, we can let Y follow a Bernoulli distribution with parameter π. This indicates that for each observation, it has a probability π_0 of being generated from Poisson(μ_0) and a probability $\pi_1 = 1 - \pi_0$ of being generated from Poisson(μ_1).
Thus, the complete-data likelihood could be written as:
$P(Z|\theta) = P(X, Y|\theta) = P(Y|X, \theta)P(X|\theta)$, where $\theta = (\mu_0, \mu_1, \pi_0, \pi_1)$.

(2)
E-step
Suppose at iteration $t - 1$,

$$\begin{aligned}
Q(\theta, \theta^{t-1}) &= E[logP(X, Y|\theta)|X, \theta^{(t-1)}] \\
&= \sum_{k=0}^{1}\sum_{i=1}^{N} P[Y_i = k|X_i, \theta^{(t-1)}]log[\pi_k P_k(X_i|\theta_k)] \\
&= \sum_{k=0}^{1}\sum_{i=1}^{N} P[Y_i = k|X_i, \theta^{(t-1)}]log(\pi_k) + \sum_{k=0}^{1}\sum_{i=1}^{N} P[Y_i = k|X_i, \theta^{(t-1)}]log[P_k(X_i|\theta_k)] \\
&= \sum_{k=0}^{1}\sum_{i=1}^{N} P[Y_i = k|X_i, \theta^{(t-1)}]log(\pi_k) + \sum_{k=0}^{1}\sum_{i=1}^{N} P[Y_i = k|X_i, \theta^{(t-1)}]log(\frac{\mu_k^{X_i} e^{-\mu_k}}{X_i!})
\end{aligned}$$

M-step
According to E-step, the first part is a function of π_k and the second part is a function of μ_k, hence we can update π_k and μ_k separately by taking derivatives with respect to π_k and μ_k, and set them equal to 0.

Since π_k has constrain, we can build a Lagrangian based on $\lambda(\sum_{k=0}^{k} \pi_k - 1)$:

$$\frac{\partial Q(\theta, \theta^{(t-1)})}{\partial \pi_k} = \sum_{i=1}^{N} \frac{1}{\pi_k} P(Y_i = k|X_i, \theta^{(t-1)}) + \lambda = 0$$

By summing over k, we can obtain that $\lambda = -N$.
Hence,

$$\pi_k = \frac{\sum_{i=1}^{N} P(Y_i = k|X_i, \theta^{(t-1)})}{N}$$

Similarly,

$$\frac{\partial Q(\theta, \theta^{(t-1)})}{\partial \mu_k} = \sum_{i=1}^{N} (-1 + \frac{X_i}{\mu_k}) P(Y_i = k|X_i, \theta^{(t-1)}) = 0$$

Hence,

$$\mu_k = \frac{\sum_{i=1}^{N} X_i P(Y_i = k|X_i, \theta^{(t-1)})}{\sum_{i=1}^{N} P(Y_i = k|X_i, \theta^{(t-1)})}$$

Moreover, we also have:

$$\begin{aligned} P(Y_i = k|X_i, \theta^{t-1}) &= \frac{\pi_k^{(t-1)} P(X_i|\mu_k^{(t-1)})}{\sum_{k=0}^{1} \pi_k P(X_i|\mu_k^{(t-1)})} \\ &= \frac{\pi_k^{(t-1)} \mu_k^{(t-1)^{X_i}} e^{-\mu_k^{(t-1)}}}{\pi_0^{(t-1)} \mu_0^{(t-1)^{X_i}} e^{-\mu_0^{t-1}} + \pi_1^{(t-1)} \mu_1^{(t-1)^{X_i}} e^{-\mu_1^{(t-1)}}} \end{aligned}$$

Therefore, at each iteration, we will update the above μ_k and π_k.

(3)
In terms of initial estimator, we can simply obtain a random starting estimates for μ_k and

π_k. But since EM might converge to local maximum, we need to try different starting values and pick the one with the highest $LL(\hat{\theta})$.

(4)
R code implementing EM on two Poisson mixture distribution is presented in the Appendix. 3 scenarios are considered here and their results are summarized below. In each scenario, 1000 observations were randomly generated from two Poisson distributions and got mixed together based on a pre-specified rate. EM-based clustering method is then applied to estimate the true parameters. After parameter estimates, clustering assignment is based on posterior probabilities. In other words, if posterior probability of being generated from model 1 is higher than its counterpart of being generated from model 2, then this observation is assigned to model 1 cluster.
Before talking about results. Two observations need to be mentioned. Firstly, EM might sometimes converge to local maximum due to the usage of random starting values. To alleviate local convergence, I simply re-run the program multiple times and chose the one with the highest LL. Secondly, there is an identification issue due to the symmetry of clustering. For instance, as described in the scenario 1 below, we hope to uncover $\mu_0 = 2$, $\mu_1 = 12$, $\pi_0 = 0.25$, but actually we got $\mu_0 = 12$, $\mu_1 = 2$, $\pi_0 = 0.74$. This is OK if it yields good clustering results.

Scenario 1: $\mu_0 = 2$, $\mu_1 = 12$, $\pi_0 = 0.25$
By leveraging the similarity measures we used in Lab3, namely Jaccard, correlation, and matching coefficients, we can gauge the performance of clustering. Clustering results are presented in Figure 1. In this case, even we couldn't identify the true parameters due to symmetry ($\hat{\mu_0} = 12$,$\hat{\mu_1} = 2$,$\hat{\pi_0} = 0.74$), we still obtain good clustering results with Jacaard = 0.94, matching = 0.96, correlation = 0.97.

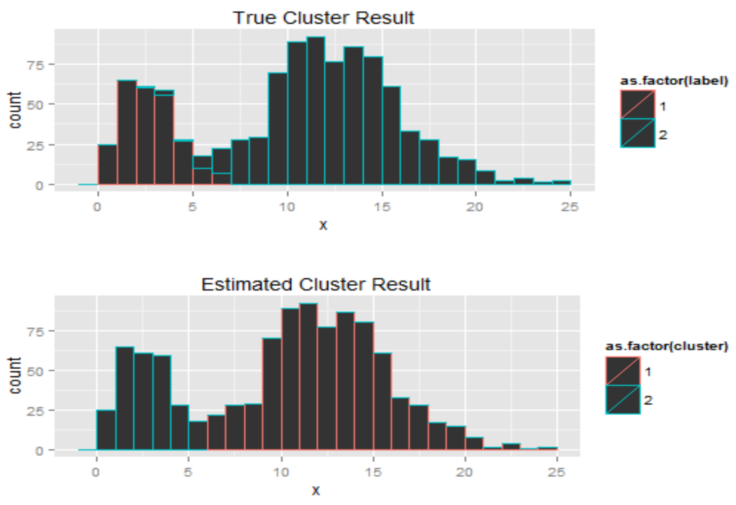

Figure 1: True Cluster Result VS Estimated Cluster Result-Scenario 1

Scenario 2: $\mu_0 = 5$, $\mu_1 = 7$, $\pi_0 = 0.2$
In this scenario, we select two parameters with closer values. In this case, even we can uncover the true parameters ($\hat{\mu_0} = 4.8$,$\hat{\mu_1} = 7.0$,$\hat{\pi_0} = 0.15$), the clustering results become bad with Jacaard = 0.68, matching = 0.69, correlation = 0.83.

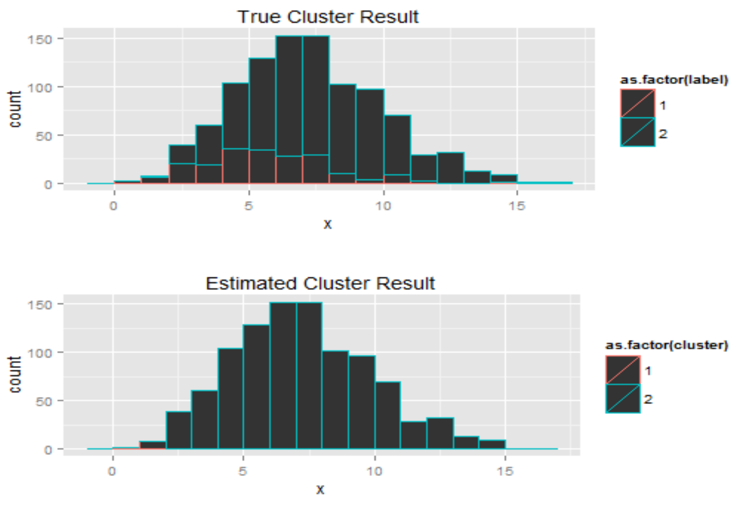

Figure 2: True Cluster Result VS Estimated Cluster Result-Scenario 2

Scenario 3: $\mu_0 = 5$, $\mu_1 = 7$, $\pi_0 = 0.4$
Same setting as scenario 2, except that I increase the mixing rate from 0.2 to 0.4. In this case, distinguishing two densities seems to be even harder. The estimated parameters are way off ($\hat{\mu_0} = 6.4$,$\hat{\mu_1} = 3.1$,$\hat{\pi_0} = 0.96$), the clustering results become worse with Jacaard = 0.52, matching = 0.52, correlation = 0.72.

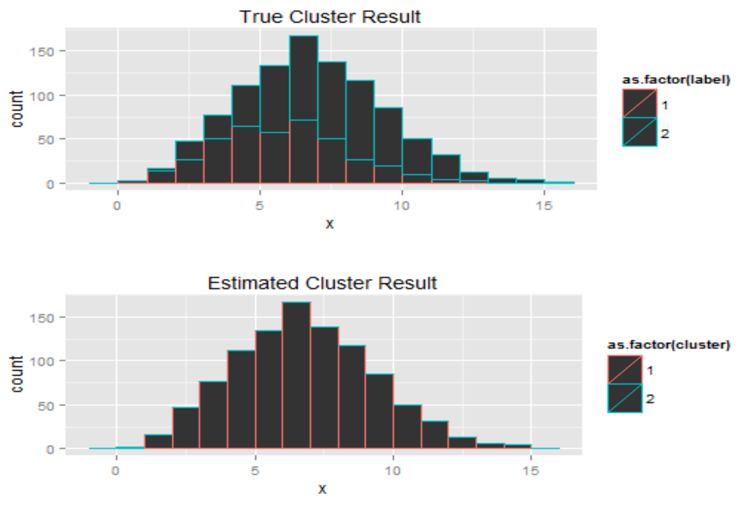

Figure 3: True Cluster Result VS Estimated Cluster Result-Scenario 3

(5)
There could be two general ways to build confidence intervals for EM estimates. The first one is to use bootstrap in which the EM algorithm is applied many times, using a different sample of the observations each time. Then the CI could be built based on the distribution of parameter estimates. However, such method should be used with cautious due to the identification issue as I pointed in Scenario 1. In this case, if we applied bootstrap, we might end up with a bi-mode distribution of one parameter. The second method to construct standard error estimates using asymptotic argument. Once the maximum of $LL(\theta)$ has been found with the EM algorithm, the standard errors can be calculated from LL the same as if the LL function had been maximized directly. The asymptotic standard errors can be calculated from the Hessian or from the variance of the observation-specific gradients(i.e., the scores), calculated from $LL(\theta)$ evaluated at $\hat{\theta}$ [1].

References

[1] Train K., 2009. Discrete Choice Methods with Simulation. Cambridge University Press, Second edition.

Appendices

```
##generate simulated data
set.seed(12345)
n<-1000
mu0_true<-5
mu1_true<-7
phi_true<-0.4
label1<-rep(1,n*phi_true)
label2<-rep(2,n*(1-phi_true))
X1<-cbind(rpois(n*phi_true,mu0_true),label1)
X2<-cbind(rpois(n*(1-phi_true),mu1_true),label2)
X<-data.frame(rbind(X1,X2))
names(X)[names(X)=="label1"] <- "label"
names(X)[names(X)=="V1"] <- "x"

options(digits=22)
.Machine$double.eps<-2.220446e-20

em_Poisson <- function(X, tol=.Machine$double.eps){
##x takes input of data.frame

  N=nrow(X) ##number of observations
  x=X$x
  error<-Inf
  iter<-1

  ##initial guess, random starts
  mu0<-sample(1:100, 1)
  mu1<-sample(1:100, 1)
  phi<- runif(1,0,1)

  while(error > tol ){

    ##E-step
    X$q_x1<-phi*dpois(x,mu0)
    X$q_x2<-(1-phi)*dpois(x,mu1)
    X$P1_x<-X$q_x1/(X$q_x1+X$q_x2) ##P=1|X
    X$P2_x<-X$q_x2/(X$q_x1+X$q_x2) ##P=2|X
    Q<-sum(log(X$q_x1)*X$P1_x)+sum(log(X$q_x2)*X$P2_x)

    ##M-step/update parameters
    mu0_k<-sum(x*X$P1_x)/sum(X$P1_x)
    mu1_k<-sum(x*X$P2_x)/sum(X$P2_x)
    phi_k<-sum(X$P1_x)/N

    ##compare Q
    X$q_x1_k<-phi_k*dpois(x,mu0_k)
    X$q_x2_k<-(1-phi_k)*dpois(x,mu1_k)
    X$P1_x_k<-X$q_x1/(X$q_x1+X$q_x2)
    X$P2_x_k<-X$q_x2/(X$q_x1+X$q_x2)
    Q_k<-sum(log(X$q_x1_k)*X$P1_x_k)+sum(log(X$q_x2_k)*X$P2_x_k)

    ##stop criterion
    error<-Q_k-Q
    iter<-iter+1
    mu0<-mu0_k
    mu1<-mu1_k
    phi<-phi_k
  }
  theta<-c(mu0,mu1,phi)

  ##cluster assingment based on posterior probability,
  ##normalizing term is canceled out
  X$cluster[phi*dpois(x,mu0)>=(1-phi)*dpois(x,mu1)]<-1
  X$cluster[phi*dpois(x,mu0)<(1-phi)*dpois(x,mu1)]<-2

  return(list(X,theta))
}

####EM-based Clustering
a<-em_Poisson(X)
##a is a list containing X with cluster assignment and parameter estimates


####calculate similarity measures, like jaccard, correlation, matching
clust_1<-as.integer(as.character(a[[1]]$label))
clust_2<-as.integer(as.character(a[[1]]$cluster))
```

```
inter.dim <- dim(X)[1]
C_1 <- matrix(clust_1, nr = inter.dim, nc = inter.dim)
        == matrix(clust_1, nr = inter.dim, nc = inter.dim, byrow = TRUE)
C_2 <- matrix(clust_2, nr = inter.dim, nc = inter.dim)
        == matrix(clust_2, nr = inter.dim, nc = inter.dim, byrow = TRUE)
diag(C_1) <- 0
diag(C_2) <- 0

jaccard <- sum(C_1 * C_2)/(sum(C_1) + sum(C_2) - sum(C_1 * C_2))
matching<- (sum(C_1 * C_2)+sum((1-C_1) * (1-C_2)))/(sum(C_1 * C_2)+
        sum((1-C_1) * (1-C_2))+sum((1-C_1)*C_2)+sum((1-C_2)*C_1))
corr<-sum(C_1 * C_2)/sqrt(sum(C_1)*sum(C_2))

print(jaccard)
print(matching)
print(corr)

####plot the clustering results
m<-ggplot(X, aes(x = x,color=as.factor(label)))
m1<-m + geom_histogram(binwidth = 1)+ggtitle("True Cluster Result")

m2<-ggplot(a[[1]], aes(x = x,color=as.factor(cluster)))
m2<-m2 + geom_histogram(binwidth = 1)+ggtitle("Estimated Cluster Result")

multiplot(m1,m2, cols=1)
```