2018 Fall

Computational Statistics
HW#4

182STG18 이하경

Description

E-M 알고리즘은 반복적인 계산을 통해 데이터가 불완전한 경우, 또는 추가적인 관측되지 않는 변수를 필요로 w 하는 경우 등에서 모수의 MLE 를 효과적으로 추정할 수 있는 방법이다. MLE 는 다른 추정치들에 비해 분산이 작으며 sample 의 크기가 커질수록 실제 값에 수렴하는 Consistency 와 Asymptotic Normality 등의 성질을 가지고 있어 MLE 의 값을 추정할 수 있다면 다른 추정치들을 사용하는 것보다 효과적이다. E-M 알고리즘은 log-likelihood 의 조건부 기댓값을 계산하는 'Expectation' Step 과 이를 최대화하는 추정치를 계산하는 'Maximization' Step 으로 이루어지며, 추정치의 값이 변하지 않을 때까지 두 Step 을 반복하여 업데이트한다.

본 과제에서는 Gaussian Mixture Model, Bivariate Normal with missing values, Regression with missing Y's, Multinomial with Complex Cell Structure 총 4 가지 예제에서 E-M 알고리즘을 각각 직접 구현해보고 최적의 MLE 를 추정해본다.

[Problem 1] GMM (Gaussian Mixture Model) – 1-dimensional & 2-class

Goal $X_i \sim (\text{iid}) \sum_{k=1}^{2} \pi_k \phi_k$으로 두 개의 정규분포 $\phi(\mu_1, \sigma_1)$, $\phi(\mu_2, \sigma_2)$의 혼합분포에서 추출된 sample 일 때 모수들과 cluster proportion π의 MLE 와 각 X_i의 class membership $\hat{Y}_i$를 E-M 알고리즘을 이용해 추정해보려고 한다.

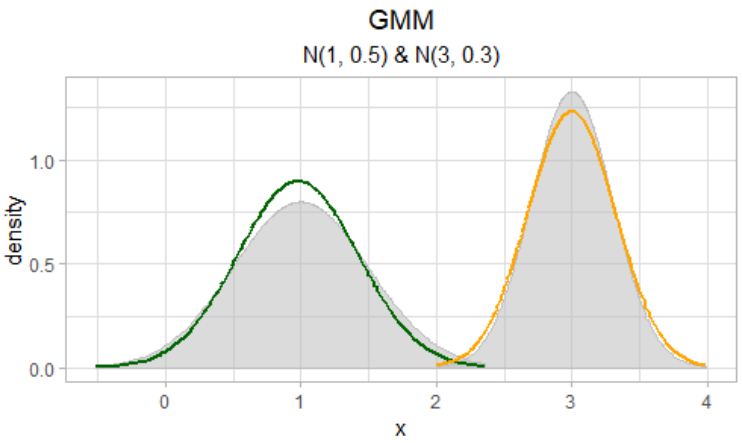
먼저 두 개의 정규분포 $\phi_1(1, 0.5)$, $\phi_2(3, 0.3)$으로 가정하고 ϕ_1에서 300 개, ϕ_2에서 700 개의 sample 을 뽑아 총 1000 개의 X_i를 생성하고 EM 알고리즘을 이용해 MLE $\hat{\theta} = (\hat{\pi}, \widehat{\mu_1}, \widehat{\mu_2}, \widehat{\sigma_1}, \widehat{\sigma_2})$과 class membership $\hat{Y}$을 추정하고 결과를 실제 값과 비교해보았다.

① 초기값 $\pi = 0.5$ 으로, μ_1과 μ_2는 Xi 중 임의로 두 개를 선택하여 지정하고 σ_1과 σ_2의 초기값으로는 X 의 sample standard deviation 으로 지정한다.
② (E-step) Bayes Theorem 을 이용하여 $\mathrm{E}[\mathrm{Y}|\theta, \mathrm{X}] = \mathrm{P}(\mathrm{Y} = 1|\theta, \mathrm{X})$을 계산한다.
③ (M-step) $\mathrm{E}[\mathrm{Y}|\theta, \mathrm{X}] \geq 0.5$일 경우 Y=1, $\mathrm{E}[\mathrm{Y}|\theta, \mathrm{X}] < 0.5$일 경우 Y=0 으로 하여 모수의 추정치를 update 한다.
④ $\text{Error} = \left| \pi^{(k)} - \pi^{(k-1)} \right| < \epsilon = 10^{-10}$을 만족할 때까지 ②와 ③을 반복한다.

Result

	$\hat{\pi}$	$\widehat{\mu_1}$	$\widehat{\mu_2}$	$\widehat{\sigma_1}$	$\widehat{\sigma_2}$
MLE	0.294	0.9784	2.9999	0.4450	0.3238
True	0.3	1	3	0.5	0.3

niter	**comp.time**
3	0.002



반복 수 3 회만에 값이 빠르게 수렴하였고, 추정된 분포의 모수들을 확인해보았을 때 실제 값과 꽤 근접하게 추정된 것을 알 수 있었다. 그래프에서 어두운 영역으로 표시된 부분은 실제 혼합 분포의 density 이며, 굵은 선으로 표시된 분포는 각각 MLE 로 추정된 모수들의 정규 분포이다. ϕ_1 은 scale 이 실제에 비해 조금 더 작게 추정되었으며,

ϕ_2는 조금 더 크게 추정되어 더 완만한 형태를 띄고 있다.

	Estimated $\hat{Y}$	
True Y	**0**	**1**
0	**699**	1
1	7	**293**

실제 가정한 class membership 과 추정된 값을 비교해볼 때 총 1000 개의 X_i 중 두 정규분포의 density 가 겹치는 구간에서의 8 개를 제외한 992 개가 올바르게 추정되었으며 오분류율은 0.8%으로 매우 작다.

[Problem 2] Bivariate Normal with Missing Values

Data w1 (complete) & w2 (missing w29, w210)

w1	8	11	16	18	6	4	20	25	9	13
w2	10	14	16	15	20	4	18	22	**NA**	**NA**

Goal 이변량 정규분포의 관측치 중 한쪽에 결측 값이 존재하는 경우, E-M 알고리즘과 이변량 정규분포의 성질을 이용해 각 결측 값의 조건부 기댓값을 계산하고 MLE ($\widehat{\mu_1}$, $\widehat{\mu_2}$, $\widehat{\sigma_{11}}$, $\widehat{\sigma_{12}}$, $\widehat{\sigma_{22}}$)을 추정한다. 이 경우 w1 은 완전 데이터이므로 $\widehat{\mu_1}$, $\widehat{\sigma_{11}}$의 값은 알고리즘 반복 시에도 변하지 않는다.

① 모수의 초기값을 각각 결측이 존재하는 관측치를 제외하고 구한 평균과 분산의 값으로 지정한다.
② (E-step) $E[w_{29}|w_{19}]$, $E[w_{210}|w_{110}]$, $E[{w_{29}}^2|w_{19}]$, $E[{w_{210}}^2|w_{110}]$ 총 4 개의 조건부 기댓값을 계산한다.
③ (M-step) 위에서 구한 기댓값을 이용해 충분통계량 T_2와 T_{12}, T_{22}를 각각 update 한다.
④ $\text{Error} = \left|\widehat{\mu_2}^k - \widehat{\mu_2}^{k-1}\right| < \epsilon = 10^{-10}$을 만족할 때까지 ②와 ③을 반복한다.

Result

	$\widehat{\mu_1}$	$\widehat{\mu_2}$	$\widehat{\sigma_{11}}$	$\widehat{\sigma_{12}}$	$\widehat{\sigma_{22}}$
MLE	13	14.6152	40.2	20.8852	26.7541

niter	**comp.time**
17	0.0136

반복 수 17 회 만에 Error 가 10^{-10} 밑으로 수렴하였다. 여러 번 반복해서 계산하였을 때 추정치가 계속 동일하였으므로 타당하게 추정되었다고 판단하였다. 따라서 추정된 $w = (w_1, w_2)$ 의 이변량 정규분포는 $N_2\left(\hat{\mu} = (13, 14.615)', \hat{\Sigma} = \begin{bmatrix} 40.2 & 20.885 \\ 20.885 & 26.754 \end{bmatrix}\right)$ 이고, 이 때 w1 과 w2 사이의 상관계수의 추정치는 $\hat{\rho} = \frac{\widehat{\sigma_{12}}}{\sqrt{\widehat{\sigma_{11}}\widehat{\sigma_{22}}}} = 0.6368$ 이다.

[Problem 3] Regression with missing Y's

Goal 회귀분석에서 반응변수 Y 에 결측(missing)이 존재하는 경우, E-M 알고리즘을 이용해 각 결측 값의 기댓값을 추정하고 최적의 회귀 모형을 찾는다.

Data. UCI Machine Learning Repository – Wine Quality Data Set (Red)

Y	Quality (Score between 0-10)	**n**	1599
X	fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol	**p**	11

총 1599 가지 포르투갈 Vinho Verde 의 레드 와인의 물리·화학적인 11 가지 속성을 값으로 가지는 설명변수들과 0 에서 10 까지의 정수형으로 된 와인 품질의 점수 데이터를 이용하였다.

먼저 1599 개의 관측치 중 10%에 해당하는 159 개의 행을 임의로 선택하여 반응변수 Quality 의 값을 결측으로 바꾸었다. 이후 다음의 순서에 따라 E-M 알고리즘을 구현하였다.

① 결측된 Y 들의 초기값으로 나머지 Y 값들의 평균인 $\bar{Y} = 5.6257$ 을 지정한다.
② (E-step) 가능한 설명변수들을 이용해 회귀 모형 $Y = f(X)$ 을 적합하여 Y 값을 예측한다. 이 경우 모형 적합에는 Stepwise Regression 방법과 Random Forest 방법을 이용하였다.
③ (M-step) 위의 모형에서 추정된 $\hat{Y}$으로 결측치였던 Y 값들을 update 한다.
④ $\hat{Y}$ 값이 변하지 않을 때까지 ②와 ③을 반복한다.

$* \text{ Error} = \text{mean} \frac{\left|\hat{Y}_i^k - \hat{Y}_i^{k-1}\right|}{\hat{Y}_i^{k-1}}$

Result 1. 모형 별 실제 MSE 와 EM 알고리즘을 이용한 최종 MSE 비교

Method	MSE		Mean sq. of $\text{true } Y - \hat{Y}$(EM)	niter
	comlplete data	final model in EM		
Stepwise	0.4174	0.3748	0.4347	5 ($\epsilon = 10^{-10}$)
Random Forest 1	0.3174	0.2878	0.2988	1000 (maxiter)
Random Forest 2	0.3174	0.2884	0.3062	30 ($\epsilon = 7 \cdot 10^{-3}$)

먼저 159 개의 Y 를 missing 처리하지 않은 원래 데이터로 stepwise regression 과 random forest 모형을 적합한 결과를 저장하고, missing 처리 후 E-M 알고리즘을 이용해 추정한 결과와 비교하였다. Random Forest 의 경우 tree 의 수는 500 개로 설정하였다.

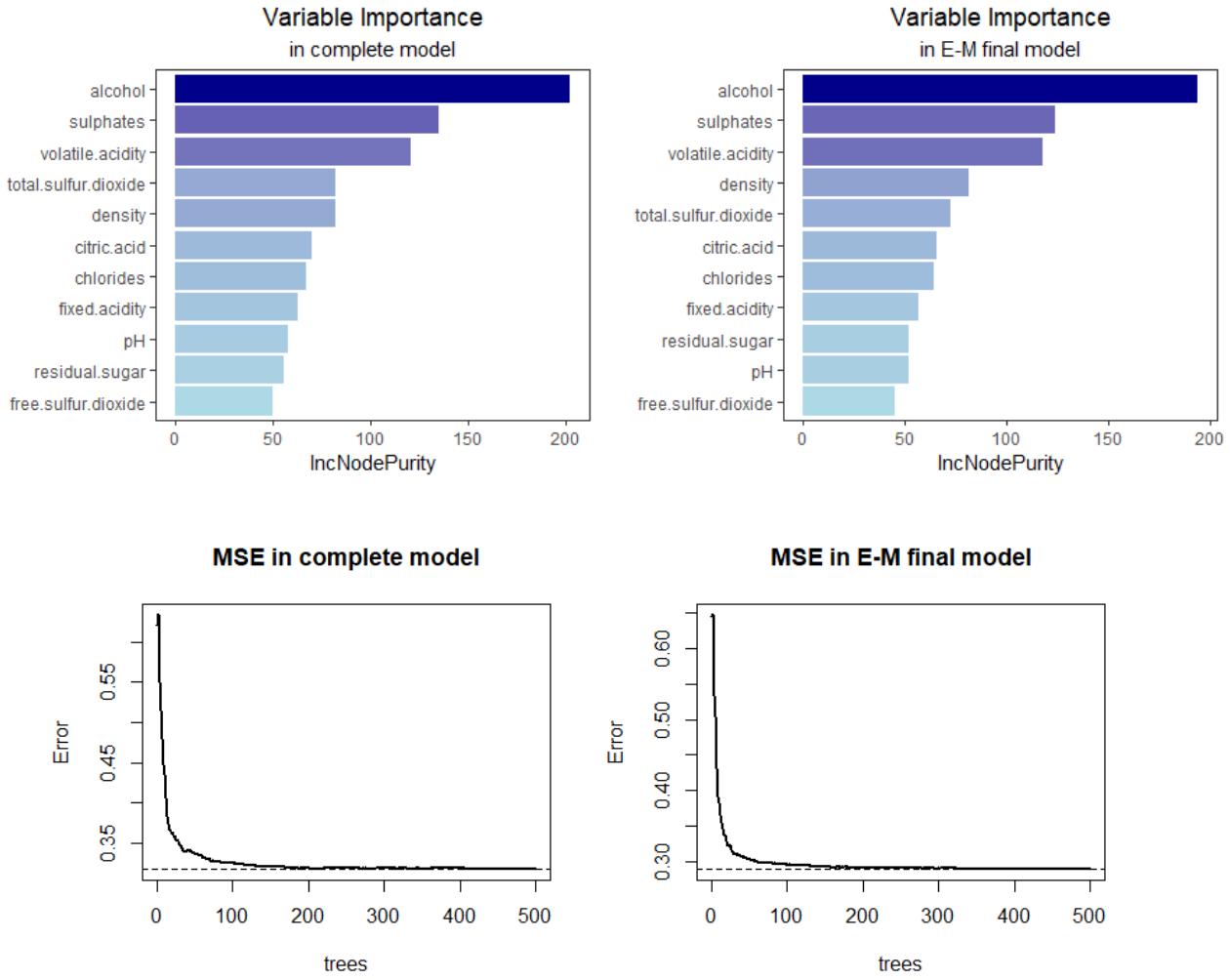
전체적으로 Stepwise 모형에서의 MSE 보다 Random Forest 모형의 MSE 가 더 작게 계산되었다. 각 방법별로 E-M 알고리즘을 거친 결과, 알고리즘을 통해 추정된 각 결측치들의 최종 $\hat{Y}$ 과 실제 Y 값의 차이의 제곱의 평균을 확인해보면 실제 MSE 보다 감소된 값을 확인할 수 있다. stepwise 의 경우 반복 수 5 회 만에 Error 가 10^{-10} 보다 작은 값으로 수렴하였다. 랜덤 포레스트의 경우 초기 threshold $\epsilon = 10^{-10}$ 으로 하였을 때 최대 반복 수 1000 에 도달할 때까지 Error 가 0.007 밑으로 떨어지지 않았다. 하지만 반복 수 3 회부터 모든 Error 가 0.007 에서 0.008 을 벗어나지 않는 것을 확인하고 threshold 값을 0.007 로 조정한 뒤 결과를 다시 한 번 확인하였을 때 최대 반복 수에 도달한 결과와 큰 차이는 발생하지 않았다.

Result 2. Stepwise Regression 의 결과: 선택된 변수들의 회귀 계수와 R^2

	complete	EM		complete	EM
(Intercept)	4.4301	4.1010	**total sulfur dioxide**	-0.0034	-0.0031
alcohol	0.2893	0.2970	**chlorides**	-2.0178	-1.9794
volatile acidity	-1.0128	-1.0084	**pH**	-0.4826	-0.4011
sulphates	0.8827	0.7980	**free sulfur dioxide**	0.0051	0.0047
			AIC	-1380.79	-1553.01
			R^2	36.0%	38.1%

추정된 값들을 확인해보면 실제 Y 값을 이용했을 때와 결측을 가정하고 E-M 알고리즘을 이용해 추정한 Y 의 추정값을 입력했을 때의 결과가 크게 다르지 않다. 최종 모형에서 선택된 변수의 개수는 7 개로 변수의 종류 또한 동일하였으며 AIC 와 R^2 값 모두 실제보다 더 좋은 결과를 나타냈다. 따라서 결측치가 존재하는 경우에도 큰 문제없이 E-M 알고리즘을 이용해 효과적인 추정을 할 수 있음을 알 수 있다.

Result 3. Random Forest 의 결과: 변수 별 중요도와 MSE



Random Forest 방법은 반복수가 최대에 도달할 때까지 수렴한 Error 값이 작지는 않았지만 이 경우 역시 실제 모형과 결과가 거의 동일하였다. 변수의 중요도를 살펴보면 상위 5 개의 변수 중 중요도가 크게 차이 나지 않는 density 와 total sulfur dioxide 의 순서를 제외하고 값들이 거의 비슷한 것을 확인할 수 있다. MSE 의 그래프 역시 거의 동일한 결과를 보이고 있다.

[Problem 4] Multinomial with Complex Cell Structure

	frequency	prob.
O	176	r^2
A	182	p^2+2pr
B	60	q^2+2qr
AB	17	$2pq$
Total	435	

Goal 혈액형 별 빈도수 자료를 이용해 A, B, O 유전자(gene)에서 $\mathrm{P(A)=p, P(B)=q, P(O)=r}$ 의 MLE $\hat{\theta}=(\hat{p},\hat{q},\hat{r})$ 을 추정하려고 한다.

$$\log-\text{likelihood } l(\theta)$$
$$=2n_O logr+n_A\log(p^2+2pr)+n_B\log(q^2+2qr)+n_{AB}\log(2pq) \;\rightarrow\; \frac{\partial(\theta)}{\partial\theta}=0$$

직접 계산할 수 없는 p, q, r 의 MLE 를 계산하기 위해 Comeplete Cell Structure 에서 알려지지 않은 $n_{AA}, n_{AO}, n_{BB}, n_{BO}$ 의 값을 E-M 알고리즘을 이용하여 조건부 기댓값을 계산하고 MLE 를 추정한다.

	frequency	prob.
O	176	r^2
AA	**(unobserved)**	p^2
AO	**(unobserved)**	$2pr$
BB	**(unobserved)**	q^2
BO	**(unobserved)**	$2qr$
AB	17	$2pq$
Total	435	

$$lc(\theta)=2n_A{}^+logp+2n_B{}^+logq+2n_O{}^+logr$$
$$*\,n_A{}^+=n_{AA}+\frac{1}{2}n_{AO}+\frac{1}{2}n_{AB}, n_B{}^+=n_{BB}+\frac{1}{2}n_{BO}+\frac{1}{2}n_{AB}, n_O{}^+=n_O+\frac{1}{2}n_{AO}+\frac{1}{2}n_{BO}$$

($\mathrm{Log-Likelihood\ of\ Multinomial(p,q,r)}$with frequency $2n_A{}^+, 2n_B{}^+, 2n_O{}^+$ 의 형태로 계산할 수 있음)

① unobserved frequency 의 초기값을 각각 $n_{AA}, n_{AO}=0.5n_A,\ n_{BB}, n_{BO}=0.5n_B$로 지정하여 p, q, r 의 초기값을 계산한다.
② (E-step) $\mathrm{E}[n_{AA}|\,n_A]$, $\mathrm{E}[n_{AO}|\,n_A]$, $\mathrm{E}[n_{BB}|\,n_B]$, $\mathrm{E}[n_{BO}|\,n_B]$를 각각 계산한다.
③ (M-step) $\hat{p}=\frac{n_A{}^+}{n_A}, \hat{q}=\frac{n_B{}^+}{n_B}, \hat{r}=1-\hat{p}-\hat{q}$의 값을 update 한다.
④ $\hat{p}$ 의 값이 변하지 않을 때까지 반복한다.

Result

$\hat{p}$	$\hat{q}$	$\hat{r}$	niter	comp.time
0.2704	0.0940	0.6356	11	0.0107

반복 수 11 회 만에 값이 빠르게 수렴하였다. 추정된 $\mathrm{P(A), P(B), P(O)}$ 의 값을 확인해볼 때 $\mathrm{P(O)>P(A)>P(B)}$ 의 순서로 값이 크다. 따라서 주어진 A, B, AB, O 의 Cell Structure 에서 O 유전자의 출현빈도가 가장 높으며 B 유전자의 출현빈도가 가장 낮음을 예상할 수 있다.

Discussion

위의 4 가지 예제를 통해 다양한 E-M 알고리즘을 R 에서 실제로 구현해보고 추정 값을 구하였다. 2 번과 3 번 예제와 같이 데이터에 결측이 존재하는 경우에도 문제없이 추정이 가능하였고, 특히 실제 데이터가 확인 가능했던 3 번의 경우는 결측을 가정하지 않고 실제 데이터를 이용하였을 때의 예측 회귀 모형의 결과와 E-M 알고리즘을 이용한 결과가 거의 동일하였다. 1 번과 4 번의 예제처럼 MLE 계산을 위해 알려지지 않은 값의 존재를 가정하고 계산해야 하는 경우에도 안정적으로 MLE 의 값을 추정할 수 있었다. 따라서 위와 같이 log-likelihood 를 최대화하는 MLE 를 직접적으로 계산할 수 없을 때 Expectation 과 Maximization 의 반복적인 계산을 통해 이를 추정할 수 있도록 하는 효과적인 알고리즘이라는 것을 깨달을 수 있었다.

[Appendix] R Code

```
# 1) GMM
  mygmm <- function(x, E=10^(-10))
  {
    niter = 0 ; maxiter = 1000 ; error = 1

  # initial values
    n = length(x)
    prop = 0.5
    set.seed(0)
    mu = sort(sample(x, 2)) ; mu1 = mu[1] ; mu2 = mu[2]
    sd1 = sd(x) ; sd2 = sd(x)

  # E-M
    while (error >= E & niter <= maxiter)
    { prop0 <- prop ; mu10 <- mu1 ; mu20 <- mu2 ; sd10 <- sd1 ; sd20 <- sd2
      Ey = c() ; y = c()
      for (i in 1:n)
      { Ey[i] <- prop0*dnorm(x[i], mu10, sd10) / ( prop0*dnorm(x[i], mu10, sd10) + (1-prop0)*dnorm(x[i], mu20, sd20) )
        y[i] <- round(Ey[i])
      }

      prop <- sum(y)/n
      mu1 <- sum(y*x)/sum(y) ; mu2 <- sum((1-y)*x)/sum(1-y)
      sd1 <- sqrt(sum(y*(x-mu1)^2)/sum(y)) ; sd2 <- sqrt(sum((1-y)*(x-mu2)^2)/sum(1-y))

      error <- abs(prop - prop0)
      niter <- niter + 1

      print(paste("error =", error, "niter = ", niter, sep = " "))
    }

    output <- list(y=y, prop=prop, mu1=mu1, mu2=mu2, sd1=sd1, sd2=sd2)
    return(output)
  }

  set.seed(100)
  x1 <- c(rnorm(300, mean = 1, sd = 0.5), rnorm(700, mean = 3, sd = 0.3)) # (true value) prop=0.3 ; mu1=1 ; sd1=0.5 ; mu2=3 ; sd2=0.3
  ex1 <- mygmm(x1) ; ex1

  system.time( for (i in 1:10) mygmm(x1, E=10^-10) )

  ggplot() + theme_light() + labs(x = "x", y = "density", title = "GMM", subtitle = "N(1, 0.5) & N(3, 0.3)") +
    theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5), legend.position = "") +
    scale_color_manual("", values = c(real = "gray", n1 = "darkgreen", n2 = "orange")) +
    scale_fill_manual("", values = c(real = "gray", n1 = "darkgreen", n2 = "orange")) +
    geom_area(aes(x1[1:300], dnorm(x1[1:300], 1, 0.5), fill = "real", alpha = 0.5, color = "real")) +
    geom_area(aes(x1[301:1000], dnorm(x1[301:1000], 3, 0.3), fill = "real", alpha = 0.5, color = "real")) +
    geom_line(aes(x1[1:300], dnorm(x1[1:300], ex1$mu1, ex1$sd1), color = "n1"), size = 1) +
    geom_line(aes(x1[301:1000], dnorm(x1[301:1000], ex1$mu2, ex1$sd2), color = "n2"), size = 1)

  true.y = c(rep(1, 300), rep(0, 700))
  table(true.y, ex1$y)


# 2) bivariate normal w/ missing values
```

```
  data2 <-data.frame(w1 = c(8, 11, 16, 18, 6, 4, 20, 25, 9, 13),
                     w2 = c(10, 14, 16, 15, 20, 4, 18, 22, NA, NA))

  myem2 <- function(E=10^(-10))
  {
    w1 = data2$w1 ; w2 = data2$w2 ; n = nrow(data2)

  # initial values
    T1 = sum(w1) ; T2 = sum(w2[1:8])
    T11 = sum(w1*w1) ; T12 = sum(w1[1:8]*w2[1:8]) ; T22 = sum(w2[1:8]*w2[1:8])

    mu1 = T1/n ; mu2 = T2/n
    sig11 = (T11-T1*T1/n) / n ; sig12 = (T12-T1*T2/n) / n ; sig22 = (T22-T2*T2/n) / n
    rho = sig12 / sqrt(sig11*sig22)

    niter = 0
    maxiter = 1000
    error = 1

    while (error >= E & niter <= maxiter)
    { T12_0 <- T12 ; T22_0 <- T22
      mu2_0 <- mu2 ; sig12_0 <- sig12 ; sig22_0 <- sig22 ; rho_0 <- rho
    # E-step
      Ew29 <- mu2 + sig12/sig11*(w1[9]-mu1)
      Ew29sq <- Ew29^2 + sig22*(1-rho^2)
      Ew210 <- mu2 + sig12/sig11*(w1[10]-mu1)
      Ew210sq <- Ew210^2 + sig22*(1-rho^2)

      T2 <- sum(w2[1:8]) + Ew29 + Ew210
      T12 <- sum(w1[1:8]*w2[1:8]) + w1[9]*Ew29 + w1[10]*Ew210
      T22 <- sum(w2[1:8]*w2[1:8]) + Ew29sq + Ew210sq

    # M-step
      mu2 = T2/n
      sig12 = (T12-T1*T2/n) / n ; sig22 = (T22-T2*T2/n) / n
      rho = sig12 / sqrt(sig11*sig22)

      error <- abs(mu2 - mu2_0)
      niter <- niter + 1

      print(paste("error =", error, "niter = ", niter, sep = " "))
    }

    output = data.frame(Ew29, Ew210, mu1, mu2, sig11, sig12, sig22)
    return(output)
  }

  myem2(E=10^-10)
  system.time( for (i in 1:100) myem2() )

# 3) Reg. model w/ missing Y's
  red <- read.csv("D:/STAT/CS/HW4/winequality-red.csv", sep = ";")
  white <- read.csv("D:/STAT/CS/HW4/winequality-white.csv", sep = ";")

  step.true <- step(lm(quality ~ 1, red), direction = "both",
                    scope = list(lower=lm(quality ~ 1, red), upper=lm(quality ~ ., red)))
  summary(step.true)
```

```
  rf.true <- randomForest(quality ~ ., red) ; rf.true

  set.seed(100)
  empty <- sort(sample(nrow(red), nrow(red)*0.1))
  true.y <- red[empty, 12]
  red[empty, 12] <- NA

# stepwise reg.
  myem3.1 <- function(E=10^(-10))
  {
  # initialize
    red[empty, 12] <- mean(as.numeric(red[, 12]), na.rm = T)

    niter = 0
    maxiter = 1000
    error = 1

    while(error >= E & niter <= maxiter)
    { yhat_0 <- red[empty, 12]
    # model fitting
      fit <- step(lm(quality ~ 1, red), direction = "both", trace = 0,
                  scope = list(lower=lm(quality ~ 1, red), upper=lm(quality ~ ., red)))
      yhat <- fitted(fit)[empty]
      red[empty, 12] <- yhat

      error <- mean( abs(yhat-yhat_0) / yhat_0 )
      niter <- niter + 1

      print(paste("error =", error, "niter = ", niter, sep = " "))
    }

    return(list(fit = fit, yhat = yhat))
  }

  ex3.1 <- myem3.1()
  summary(ex3.1$fit)
  mean((ex3.1$yhat - true.y)^2)


  red[empty, 12] <- NA

  # random forest
  myem3.2 <- function(E=10^(-10))
  {
  # initial
    red[empty, 12] <- mean(as.numeric(red[, 12]), na.rm = T)

    niter = 0
    maxiter = 1000
    error = 1

    while(error >= E & niter <= maxiter)
    { yhat_0 <- red[empty, 12]

     # model fitting
      set.seed(100)
      fit <- randomForest(quality ~ ., red)
```

```
    yhat <- fit$predicted[empty]
    mse <- min(fit$mse)
    red[empty, 12] <- yhat

    error <- mean( abs(yhat-yhat_0) / (yhat_0) )
    niter <- niter + 1

    print(paste("error =", error, "niter = ", niter, sep = " "))
  }

  return(list(fit = fit, yhat = yhat))
}

ex3.2 <-myem3.2()
ex3.2 <- myem3.2(0.007)
mean((ex3.2$yhat-true.y)^2)

par(mfrow = c(1, 2))
plot(rf.true, lwd = 2, main = "MSE in complete model")
abline(h = min(rf.true$mse), lty = 2)
plot(ex3.2$fit, lwd = 2, main = "MSE in E-M final model")
abline(h = min(ex3.2$fit$mse), lty = 2)

importance <- data.frame(variable = rownames(ex3.2$fit$importance), IncNodePurity = ex3.2$fit$importance)
grf2 <- ggplot(importance, aes(reorder(variable, IncNodePurity), y = IncNodePurity, fill = IncNodePurity)) +
        theme_test() + theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5), legend.position = "") +
        scale_fill_gradient(low = "lightblue", high = "darkblue") +
        coord_flip() + labs(x = "", title = "Variable Importance", subtitle = "in E-M final model") +
        geom_bar(stat = "identity", position = "dodge") ; grf2

importance <- data.frame(variable = rownames(rf.true$importance), IncNodePurity = rf.true$importance)
grf1 <- ggplot(importance, aes(reorder(variable, IncNodePurity), y = IncNodePurity, fill = IncNodePurity)) +
  theme_test() + theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5), legend.position = "") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  coord_flip() + labs(x = "", title = "Variable Importance", subtitle = "in complete model") +
  geom_bar(stat = "identity", position = "dodge") ; grf1

grid.arrange(grf1, grf2, ncol = 2)


# 4) multinomial with complex cell structure
data4 <- data.frame(type = c("O", "A", "B", "AB"),
                    freq = c(176, 182, 60, 17))

myem4 <- function(E=10^(-10))
{
  n = 435
  no = 176 ; na = 182 ; nb = 60 ; nab = 17

  # initial values
  naa = 0.5*na ; nao = 0.5*na ; nbb = 0.5*nb ; nbo = 0.5*nb
  p = (naa + 0.5*nao + 0.5*nab) / n ; q = (nbb + 0.5*nbo + 0.5*nab) / n ; r = 1-p-q

  niter = 0
  maxiter = 1000
  error = 1
```

```
  while (error >= E & niter <= maxiter)
  { p_0 <- p ; q_0 <- q
  # E-step
    naa <- na*(p^2/(p^2+2*p*r)) ; nao <- na*(2*p*r/(p^2+2*p*r))
    nbb <- nb*(q^2/(q^2+2*q*r)) ; nbo <- nb*(2*q*r/(q^2+2*q*r))
  # M-step
    p = (naa + 0.5*nao + 0.5*nab) / n ; q = (nbb + 0.5*nbo + 0.5*nab) / n

    error <- abs(p - p_0)
    niter <- niter + 1

    print(paste("error =", error, "niter = ", niter, sep = " "))
  }

  output <- data.frame(p, q, r = 1-p-q)
  return(output)
}

myem4()
system.time( for (i in 1:100 ) myem4() )
```