

SMM implementation 2

Chanwoo Lee, April 17, 2020

1 SMM implementation

1.1 Simulation 1

I generate random matrix $X_1, \dots, X_{100} \in \mathbb{R}^{5 \times 4}$ whose entries are from i.i.d. $\text{Unif}(-1, 1)$. I set ground truth matrix $B = UV^T \in \mathbb{R}^{5 \times 4}$ such that $U \in \mathbb{R}^{5 \times 3}, V \in \mathbb{R}^{4 \times 3}$ whose entries are from i.i.d. $\text{Unif}(-1, 1)$. Our classification rule is

$$y = \text{sign}(\langle B, X \rangle + 0.1).$$

First, I perform five folded cross validation with the new algorithm (multiple initialization method). Table 1 shows the miss classification rate (MCR) of SVM method and SMM method at each test. Table 2 shows the loss function value at the last iteration at each test.

	1st	2nd	3rd	4th	5th	average
SVM	0.90	0.95	0.9	0.75	0.9	0.88
SMM	0.95	0.95	1.0	0.75	0.9	0.91

Table 1: Miss classification rate in Simulation 1

	1st	2nd	3rd	4th	5th	average
SVM	21.05690	21.48122	20.0643	13.47480	11.44244	17.50393
SMM	23.06968	22.65985	27.2126	13.93015	13.09969	19.99439

Table 2: Loss function value in Simulation 1

Secondly, I use PCA method to visualize matrix in 2 dimension. I vectorize feature matrix X_i and make the design matrix,

$$X = (\text{Vec}(X_1)^T, \dots, \text{Vec}(X_N)^T)^T \in \mathbb{R}^{N \times mn}.$$

Through PCA method, I plot matrices (X_1, \dots, X_N) in terms of PC1 vs PC2 and PC1 vs PC3. In addition, I make grids as $a \cdot \text{PC1} + b \cdot \text{PC2}$ where a and b are some constant between minimum and maximum value of corresponding principal component of the data. I compare the true, SVM, and SMM boundary in the plots. Figure 1 shows the result.

1.2 Simulation 2

I generate random matrix and true label $(X_1, y_1), \dots, (X_{100}, y_{100}) \in \mathbb{R}^{5 \times 4}$ with the following rule

$$\{(X_i, 1) : X_i = U_1 V_1^T + E_i \quad i = 1, \dots, 100\} \text{ and } \{(X_j, -1) : X_j = U_2 V_2^T + E_i \quad j = 101, \dots, 200\},$$

where $U_1, U_2 \in \mathbb{R}^{5 \times 3}, V_1, V_2 \in \mathbb{R}^{4 \times 3}$ whose entries are from i.i.d. $\text{unif}(-1, 1)$, and E_i are Gaussian random matrix whose entries are from i.i.d. $N(0, 2^2)$. In this case, there is no specific ground

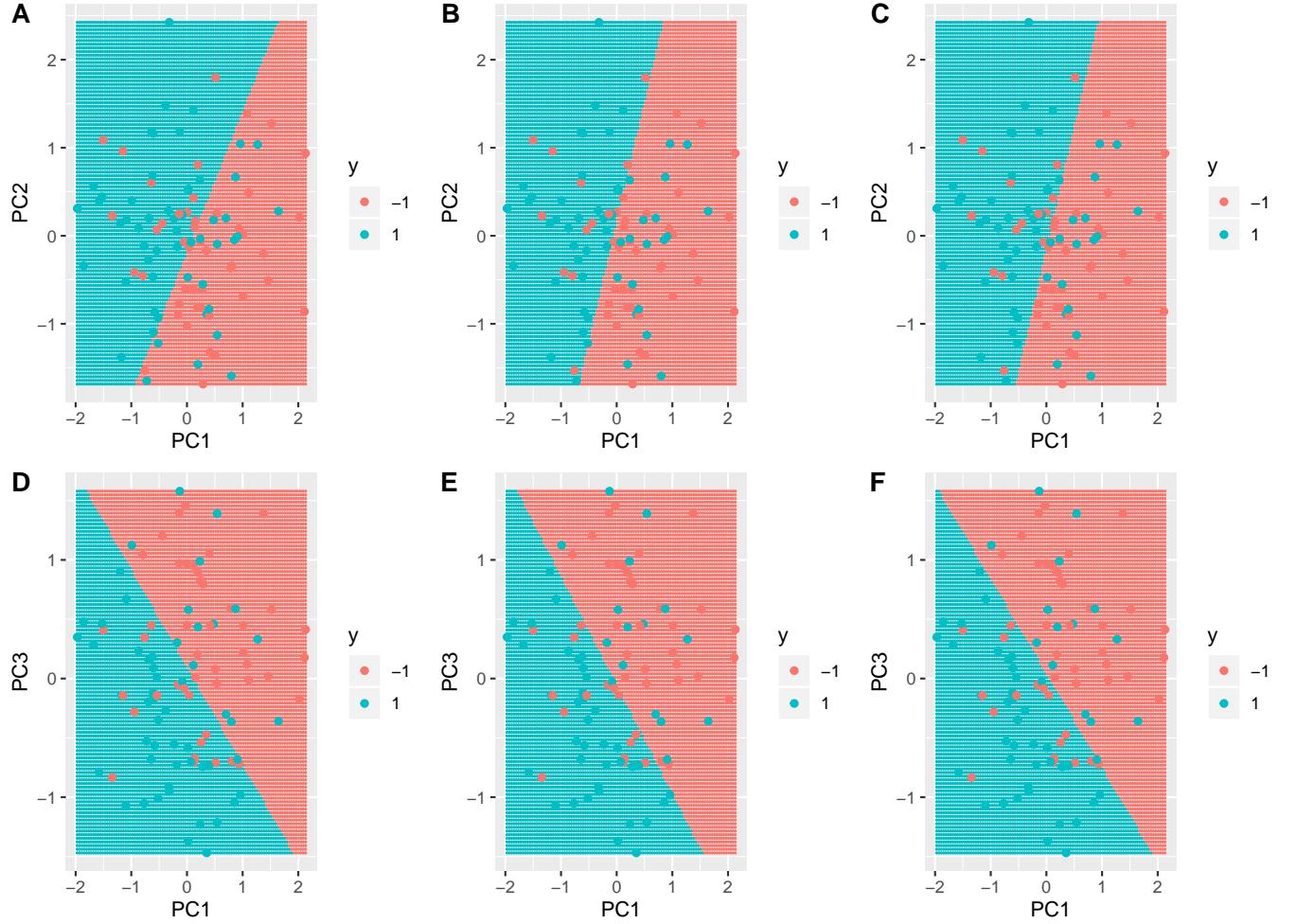


Figure 1: (A,D) are true boundary, (B,E) are SVM boundary, and (C,F) are SMM boundary. (A,B,C) are plotted with PC1 and PC2 axis and (D,E,F) are plotted with PC1 and PC3.

truth rank because this model does not have linear classification rule (It has different distribution according to label). I set rank as three to fit SMM.

I perform five folded cross validation with the new algorithm (multiple initialization method). Table 3 shows the miss classification rate (MCR) of SVM method and SMM method at each test. In SMM method, I set the rank as three. Table 4 shows the loss function value at the last iteration at each test.

	1st	2nd	3rd	4th	5th	average
SVM	0.55	0.65	0.75	0.6	0.75	0.66
SMM	0.50	0.80	0.75	0.7	0.75	0.70

Table 3: Miss classification rate in Simulation 2

Secondly, by the same way in the previous section, I plot matrices with PCA method and compare SVM and SMM classification boundary. Figure 2 shows the result.

	1st	2nd	3rd	4th	5th	average
SVM	194.3430	304.666	233.5439	275.6089	278.5445	257.3412
SMM	201.4427	322.057	233.1719	293.3010	293.5649	268.7075

Table 4: Loss function value in Simulation 2

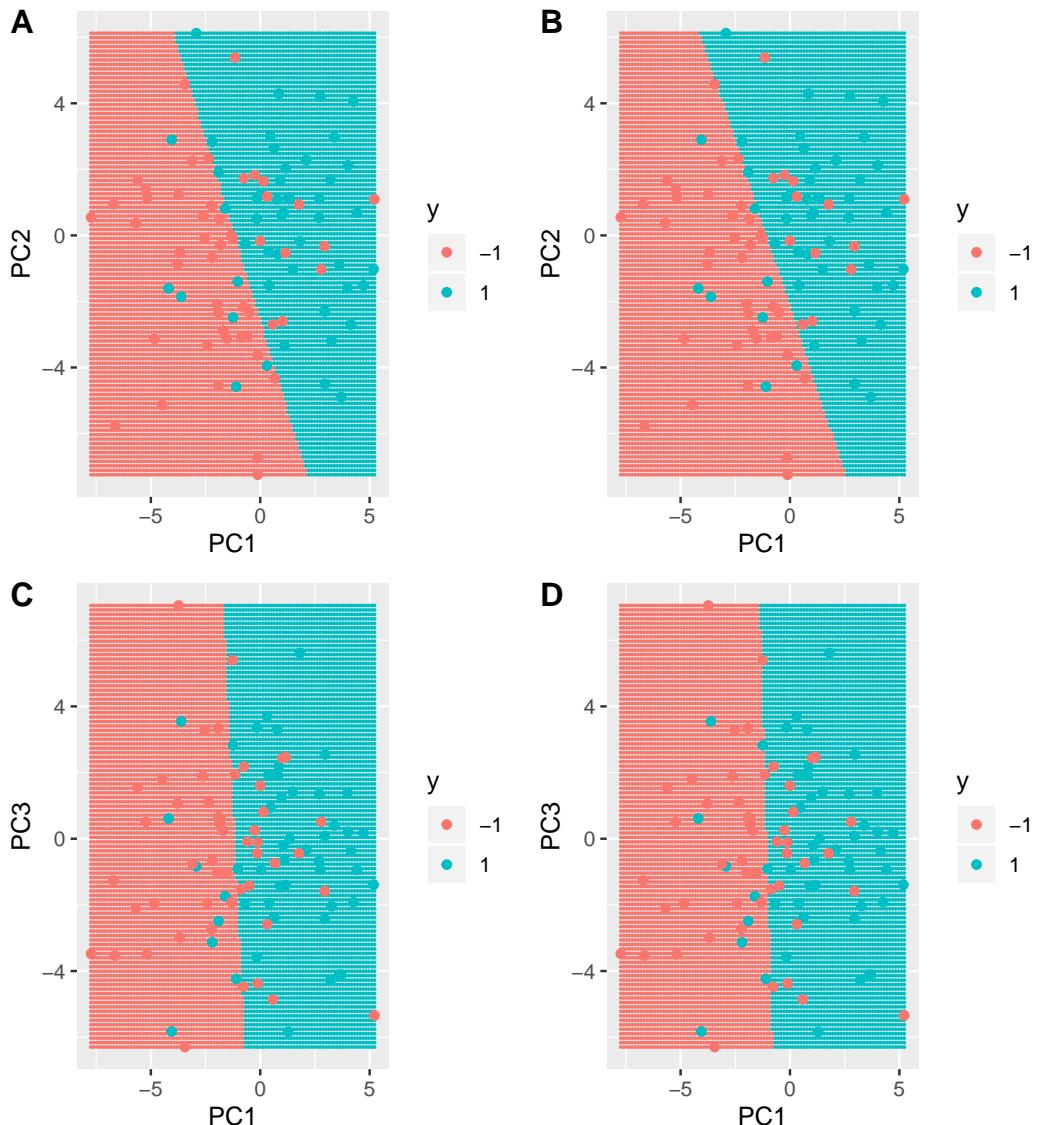


Figure 2: (A,B) are plotted with PC1 and PC2 axis and (C,D) are plotted with PC1 and PC3 axis. (A,C) and (B,D) show SVM and SMM classification boundary respectively

2 Simulation codes

```
1 source("SMMfunctions.R")
2
3 ##########
4 ## CV
5 #########
6 ### simulation 1
7 ### data generation
8 library(kableExtra)
9 set.seed(1818)
10 m = 5; n = 4; r = 3; N = 100; b0 = 0.1
11 result = gendat(m,n,r,N,b0)
12 X = result$X; y = result$y; dat = result$dat
13 B = result$B
14
15
16 ### CV
17 ind = sample(100,100)
18
19 cvresult = matrix(nrow = 2, ncol = 5)
20 objresult = matrix(nrow = 2,ncol = 5)
21 for (i in 1:5) {
22   testind = ind[((20*(i-1)+1):(20*i))]
23
24   trX = X[-testind]
25   result = svm(trX,y[-testind])
26   cvresult[1,i] = length(which(unlist(lapply(X[testind],result$predict))==y[
27     testind]))/20
28   objresult[1,i] = result$obj
29   result = smm(trX,y[-testind],3)
30   cvresult[2,i] = length(which(unlist(lapply(X[testind],result$predict))==y[
31     testind]))/20
32   objresult[2,i] = objv(result$B,result$b0,trX,y[-testind])
33   print(paste(i,"-th is done"))
34 }
35 cv = cbind(cvresult,apply(cvresult,1,mean))
36 colnames(cv) = c("1st","2nd","3rd","4th","5th","average")
37 rownames(cv) = c("SVM","SMM")
38 kable(cv, "latex")
39 cvobj = cbind(objresult,apply(objresult,1,mean))
40 colnames(cvobj) = c("1st","2nd","3rd","4th","5th","average")
41 rownames(cvobj) = c("SVM","SMM")
42 kable(cvobj, "latex")
43
44
45 #########
46 ## pca plotting
47 #########
48
49 ## data generation
50
51 result1 = svm(X,y)
52 result2 = smm(X,y,3)
53 x = dat[,-1]
54
```

```

56 library(ggplot2)
57 coord = eigen(cov(x))$vectors
58 fcoef = t(coord) %*% t(x)
59 rownames(fcoef) = paste("PC", 1:(m*n), sep = " ")
60 ndat = as.data.frame(t(rbind(y, fcoef)))
61
62 xgrid1 = seq(min(fcoef[1,]), max(fcoef[1,]), length = 100)
63 xgrid2 = seq(min(fcoef[2,]), max(fcoef[2,]), length = 100)
64 xgrid3 = seq(min(fcoef[3,]), max(fcoef[3,]), length = 100)
65
66
67
68 ###### TRUE pc1 vs pc2 and pc1 vs pc3
69 xgrid12 = expand.grid(X1 = xgrid1, X2 = xgrid2)
70 xgrid13 = expand.grid(X1 = xgrid1, X2 = xgrid3)
71
72
73 gridX12 = lapply(1:nrow(xgrid12),
74                   function(i) matrix(coord[, c(1, 2)] %*% t(xgrid12[i,]), nrow = m, ncol =
75                                         n))
76 gridX13 = lapply(1:nrow(xgrid13),
77                   function(i) matrix(coord[, c(1, 3)] %*% t(xgrid13[i,]), nrow = m, ncol
78                                         = n))
79
80 ygrid12 = unlist(lapply(gridX12, function(x) sign(sum(B*x)+b0)))
81 ygrid13 = unlist(lapply(gridX13, function(x) sign(sum(B*x)+b0)))
82
83 totgrid12 = as.data.frame(cbind(xgrid12, ygrid12))
84 totgrid13 = as.data.frame(cbind(xgrid13, ygrid13))
85
86 library(ggpubr)
87 gt12 = ggplot(data = totgrid12, aes(x = X1, y = X2, colour = as.factor(ygrid12)))+
88   geom_point(size = .1) +
89   labs(colour = "y", x = "PC1", y = "PC2") +
90   geom_point(data = ndat, aes(x = PC1, y = PC2, colour = as.factor(y)))
91 gt12
92
93 gt13 = ggplot(data = totgrid13, aes(x = X1, y = X2, colour = as.factor(ygrid13)))+
94   geom_point(size = .1) +
95   labs(colour = "y", x = "PC1", y = "PC3") +
96   geom_point(data = ndat, aes(x = PC1, y = PC3, colour = as.factor(y)))
97 gt13
98
99 ## SVM method
100
101 Bhat1 = result1$b; b0hat1 = result1$b0
102 ygrid12 = unlist(lapply(gridX12, function(x) sign(sum(Bhat1*x)+b0hat1)))
103 ygrid13 = unlist(lapply(gridX13, function(x) sign(sum(Bhat1*x)+b0hat1)))
104
105 totgrid12 = as.data.frame(cbind(xgrid12, ygrid12))
106 totgrid13 = as.data.frame(cbind(xgrid13, ygrid13))
107
108 library(ggpubr)
109 g12 = ggplot(data = totgrid12, aes(x = X1, y = X2, colour = as.factor(ygrid12)))+geom_
110   point(size = .1) +

```

```

110   labs(colour = "y",x ="PC1",y = "PC2")+
111   geom_point(data =ndat,aes(x = PC1,y = PC2,colour = as.factor(y)))
112 g12
113
114 g13 = ggplot(data = totgrid13,aes(x = X1,y = X2,colour = as.factor(ygrid13)))+geom_
115   _point(size = .1)+
116   labs(colour = "y",x ="PC1",y = "PC3")+
117   geom_point(data =ndat,aes(x = PC1,y = PC3,colour = as.factor(y)))
118 g13
119
120
121 ##### Linear SMM estimation pc1 vs pc2 and pc1 vs pc3
122
123 Bhat2 = result2$B; b0hat2 = result2$b0
124 ygrid12 = unlist(lapply(gridX12,function(x) sign(sum(Bhat2*x)+b0hat2)))
125 ygrid13 = unlist(lapply(gridX13,function(x) sign(sum(Bhat2*x)+b0hat2)))
126
127 totgrid12 = as.data.frame(cbind(xgrid12,ygrid12))
128 totgrid13 = as.data.frame(cbind(xgrid13,ygrid13))
129
130
131 gm12 = ggplot(data = totgrid12,aes(x = X1,y = X2,colour = as.factor(ygrid12)))+
132   geom_point(size = .1)+
133   labs(colour = "y",x ="PC1",y = "PC2")+
134   geom_point(data =ndat,aes(x = PC1,y = PC2,colour = as.factor(y)))
135 gm12
136
137 gm13 = ggplot(data = totgrid13,aes(x = X1,y = X2,colour = as.factor(ygrid13)))+
138   geom_point(size = .1)+
139   labs(colour = "y",x ="PC1",y = "PC3")+
140   geom_point(data =ndat,aes(x = PC1,y = PC3,colour = as.factor(y)))
141 gm13
142
143 ggarrange(gt12,g12, gm12,gt13,g13, gm13,
144             labels = c("A", "B", "C", "D", "E", "F"),
145             ncol = 3, nrow = 2)
146
147 ######simulation 2
148 ##### new data set data generation
149
150 m = 5; n = 4; r = 3
151 u1 = matrix(runif(m*r,-1,1),nrow = m,ncol = r); v1 = matrix(runif(n*r,-1,1),nrow =
152   n,ncol = r)
153 u2 = matrix(runif(m*r,-1,1),nrow = m,ncol = r); v2 = matrix(runif(n*r,-1,1),nrow =
154   n,ncol = r)
155 X = list()
156 for (i in 1:50) {
157   X[[i]] = u1%*%t(v1)+matrix(rnorm(m*n,0,2),nrow = m,ncol = n)
158   X[[i+50]] = u2%*%t(v2)+matrix(rnorm(m*n,0,2),nrow = m,ncol = n)
159 }
160 y = c(rep(1,50),rep(-1,50))
161 x = matrix(nrow =100,ncol = m*n)
162 for(i in 1:100){
163   x[i,] = as.vector(X[[i]])
164 }
165 dat = data.frame(y = factor(y), x)

```

```

164 ###### cv#####
165
166 ind = sample(100,100)
167
168
169 cvresult = matrix(nrow = 2, ncol = 5)
170 objresult = matrix(nrow = 2, ncol = 5)
171 for (i in 1:5) {
172   testind = ind[((20*(i-1)+1):(20*i))]
173
174   trX = X[-testind]
175   result = svm(trX,y[-testind],minimization = TRUE)
176   cvresult[1,i] = length(which(unlist(lapply(X[testind],result$predict))==y[
177     testind]))/20
178   objresult[1,i] = result$obj
179   result = smm(trX,y[-testind],r)
180   cvresult[2,i] = length(which(unlist(lapply(X[testind],result$predict))==y[
181     testind]))/20
182   objresult[2,i] = objv(result$B,result$b0,trX,y[-testind])
183   print(paste(i,"th is done"))
184 }
185
186 cv = cbind(cvresult,apply(cvresult,1,mean))
187 colnames(cv) = c("1st","2nd","3rd","4th","5th","average")
188 rownames(cv) = c("SVM","SMM")
189 kable(cv,"latex")
190
191 cvobj = cbind(objresult,apply(objresult,1,mean))
192 colnames(cvobj) = c("1st","2nd","3rd","4th","5th","average")
193 rownames(cvobj) = c("SVM","SMM")
194 kable(cvobj,"latex")
195
196 ##### plotting #####
197
198 result1 = svm(X,y)
199 result2 = smm(X,y,3)
200 x = dat[,-1]
201
202 library(ggplot2)
203 coord = eigen(cov(x))$vectors
204 fcoef = t(coord)%*%t(x)
205 rownames(fcoef) = paste("PC",1:(m*n),sep = " ")
206 ndat = as.data.frame(t(rbind(y,fcoef)))
207
208 xgrid1 = seq(min(fcoef[1,]),max(fcoef[1,]),length = 100)
209 xgrid2 = seq(min(fcoef[2,]),max(fcoef[2,]),length = 100)
210 xgrid3 = seq(min(fcoef[3,]),max(fcoef[3,]),length = 100)
211
212
213 #### TRUE pc1 vs pc2 and pc1 vs pc3
214 xgrid12 = expand.grid(X1 = xgrid1,X2 = xgrid2)
215 xgrid13 = expand.grid(X1 = xgrid1,X2 = xgrid3)
216
217
218 gridX12 = lapply(1:nrow(xgrid12),
219   function(i) matrix(coord[,c(1,2)]%*%t(xgrid12[i,]),nrow = m,ncol

```

```

221   = n))
222 gridX13 = lapply(1:nrow(xgrid13),
223                   function(i) matrix(coord[,c(1,3)]%*%t(xgrid13[i,]), nrow = m, ncol
224                   = n))
225
226
227 ## SVM method
228
229 Bhat1 = result1$B; b0hat1 = result1$b0
230 ygrid12 = unlist(lapply(gridX12, function(x) sign(sum(Bhat1*x)+b0hat1)))
231 ygrid13 = unlist(lapply(gridX13, function(x) sign(sum(Bhat1*x)+b0hat1)))
232
233 totgrid12 = as.data.frame(cbind(xgrid12,ygrid12))
234 totgrid13 = as.data.frame(cbind(xgrid13,ygrid13))
235
236 library(ggpubr)
237 g12 = ggplot(data = totgrid12, aes(x = X1, y = X2, colour = as.factor(ygrid12)))+geom_
238   _point(size = .1)+
239   labs(colour = "y", x = "PC1", y = "PC2")+
240   geom_point(data =ndat, aes(x = PC1, y = PC2, colour = as.factor(y)))
241 g12
242
243 g13 = ggplot(data = totgrid13, aes(x = X1, y = X2, colour = as.factor(ygrid13)))+geom_
244   _point(size = .1)+
245   labs(colour = "y", x = "PC1", y = "PC3")+
246   geom_point(data =ndat, aes(x = PC1, y = PC3, colour = as.factor(y)))
247 g13
248
249
250
251 #### Linear SMM estimation pc1 vs pc2 and pc1 vs pc3
252
253 Bhat2 = result2$B; b0hat2 = result2$b0
254 ygrid12 = unlist(lapply(gridX12, function(x) sign(sum(Bhat2*x)+b0hat2)))
255 ygrid13 = unlist(lapply(gridX13, function(x) sign(sum(Bhat2*x)+b0hat2)))
256
257 totgrid12 = as.data.frame(cbind(xgrid12,ygrid12))
258 totgrid13 = as.data.frame(cbind(xgrid13,ygrid13))
259
260 gm12 = ggplot(data = totgrid12, aes(x = X1, y = X2, colour = as.factor(ygrid12)))+
261   geom_point(size = .1)+
262   labs(colour = "y", x = "PC1", y = "PC2")+
263   geom_point(data =ndat, aes(x = PC1, y = PC2, colour = as.factor(y)))
264 gm12
265
266 gm13 = ggplot(data = totgrid13, aes(x = X1, y = X2, colour = as.factor(ygrid13)))+
267   geom_point(size = .1)+
268   labs(colour = "y", x = "PC1", y = "PC3")+
269   geom_point(data =ndat, aes(x = PC1, y = PC3, colour = as.factor(y)))
270 gm13
271
272 ggarrange(g12, gm12,g13, gm13,
273             labels = c("A", "B", "C", "D"),
274             ncol = 2, nrow = 2)

```