HW2: Data Mining

Kristyan Dimitrov, Srividya Ganapathi, Shreyashi Ganguly, Greesham Simon, Joe Zhang

1/22/2020

Problem 1

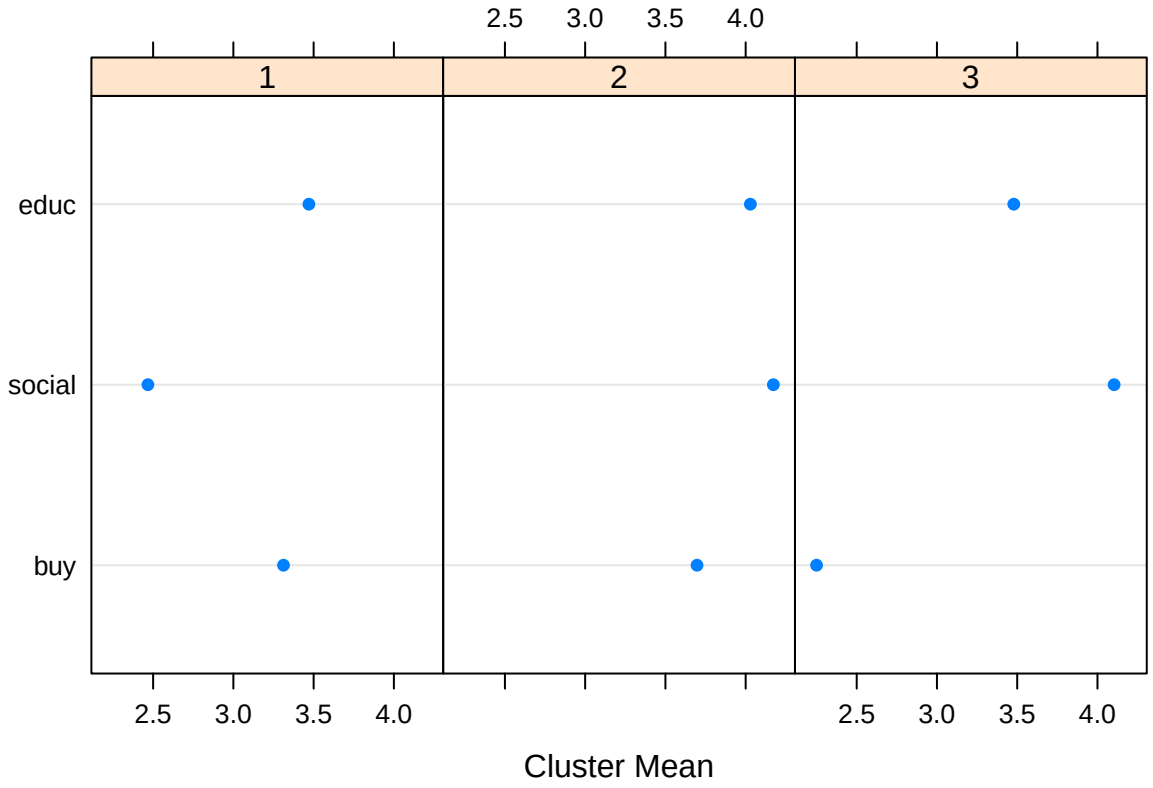
a

```
tradeshow <- read.csv('tradeshow.csv')
colnames(tradeshow) <- c("buy","social","educ")

set.seed(12345)
fit = kmeans(tradeshow, 3, 100, 100)

#cluster size, means, and RMSE included in summary
summary(fit)
```

```
##     n  Pct  buy social educ   RMSE
## 1 169 0.38 3.31   2.47 3.47 0.6076
## 2 170 0.38 3.70   4.17 4.03 0.5578
## 3 106 0.24 2.25   4.10 3.48 0.6534
##   445 1.00 3.21   3.51 3.69 0.6006
## SSE= 478.3618 ; SSB= 467.9116 ; SST= 946.2733
## R-Squared =  0.4944782
## Pseudo F =  216.1721
```

```
plot(fit)
```

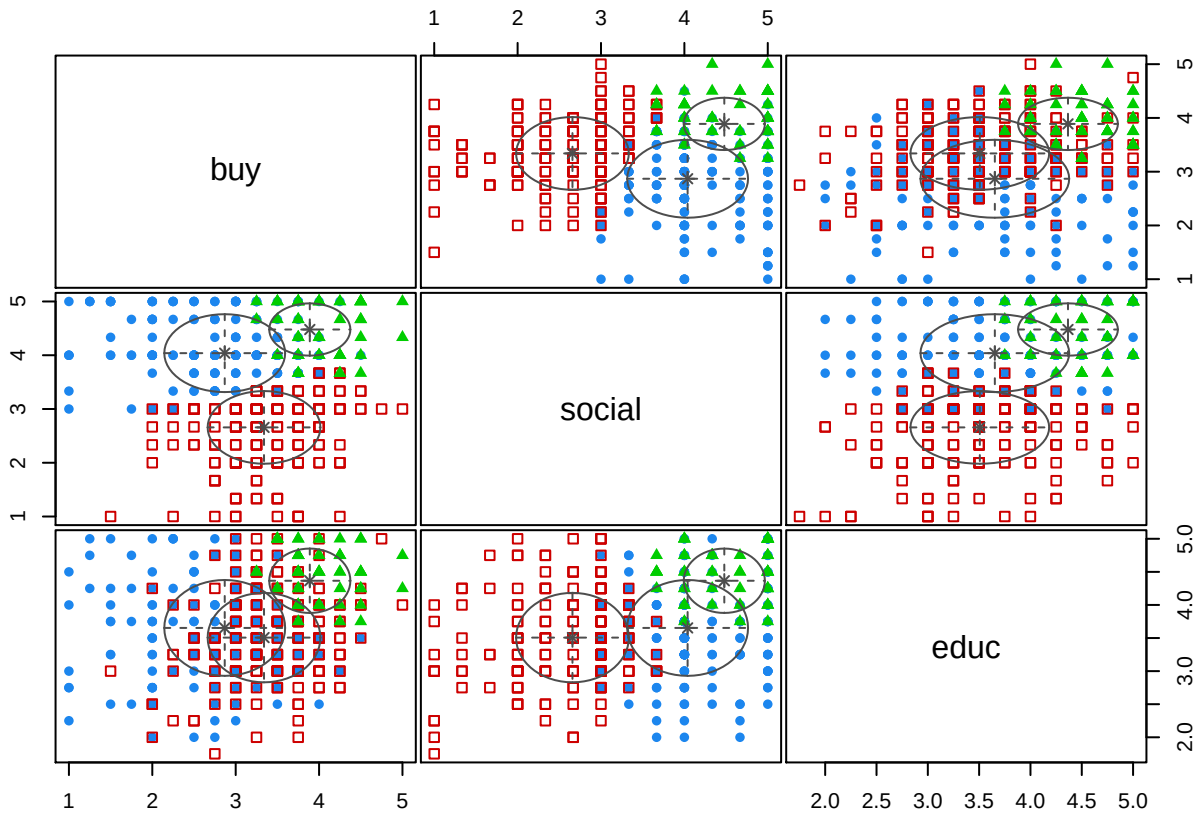
```
## Loading required package: lattice
```



Cluster Descriptions • Non social - here to educate themselves and buy some • Ambitious - here to do everything • Non buyer - here to network and educate themselves, not to buy

b

```
set.seed(12345)
fit.tds.gmm = Mclust(tradeshow,G=3,modelNames="VII")
plot(fit.tds.gmm, what = "classification")
```




```
fit.tds.gmm$parameters$pro
```

```
## [1] 0.4395187 0.4265838 0.1338976
```

```
fit.tds.gmm$parameters$mean
```

```
##             [,1]     [,2]     [,3]
## buy    2.868500 3.339815 3.888867
## social 4.038811 2.657831 4.478752
## educ   3.652959 3.506621 4.365602
```

```
sqrt(fit.tds.gmm$parameters$variance$sigmasq)
```

```
## [1] 0.7245096 0.6759707 0.4863842
```

Though the three clusters have the same descriptions, there is better distinction in their values now – Cluster 1 - Non buyer – Cluster 2 - Non social – Cluster 3 - Ambitious

K-means churned out almost equal sized clusters. However GMM has made the 'Ambitious' cluster almost one-third the size of the other two clusters or 13% of the entire sample. This solution makes more sense as there must be only a handful of 'ambitious' people, intuitively.

Both K-Means and GMM clusters have the same ordering of within cluster variances Ambitious < Non Social < Non Buyer K-Means gives very small difference in the RMSE values, GMM depicts larger differences

Number of variance parameters estimated is 3.

c

```
fit.tds.gmm1 = Mclust(tradeshow,G=3) #fitted VVE setting
```

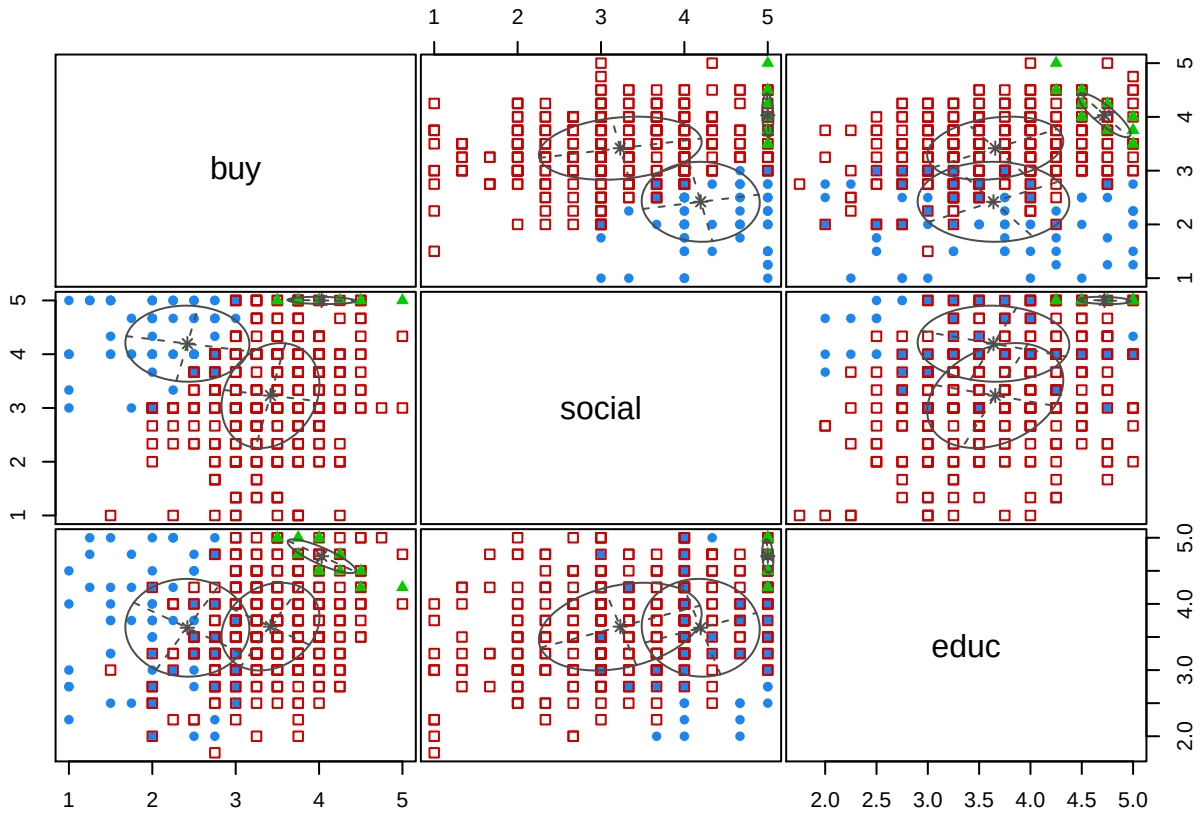
```
fit.tds.gmm1$parameters$pro
```

```
## [1] 0.23084477 0.73819961 0.03095562
```

```
fit.tds.gmm1$parameters$mean
```

```
##             [,1]     [,2]     [,3]
## buy    2.420752 3.417206 4.030997
## social 4.195782 3.231188 4.999999
## educ   3.639689 3.657019 4.721012
```

```
plot(fit.tds.gmm1, what= "classification")
```




```
fit.tds.gmm1$parameters$variance
```

```
## $modelName
## [1] "VVE"
```

```
##
## $d
## [1] 3
##
## $G
## [1] 3
##
## $sigma
## , , 1
##
##                 buy      social         educ
## buy     0.553753113 -0.01041070 -0.006476061
## social -0.010410703  0.50153507 -0.017107224
## educ   -0.006476061 -0.01710722  0.546659790
##
## , , 2
##
##               buy    social       educ
## buy    0.34510202 0.1169826 0.08611469
## social 0.11698262 0.9540970 0.19174011
## educ   0.08611469 0.1917401 0.44021197
##
## , , 3
##
##                 buy       social         educ
## buy     0.167057935 -0.003870663 -0.088292319
## social -0.003870663  0.004554370 -0.003208518
## educ   -0.088292319 -0.003208518  0.065283215
##
##
## $scale
## [1] 0.53318519 0.49734684 0.02101109
##
## $shape
##           [,1]      [,2]       [,3]
## [1,] 1.0457755 0.5916538 10.3798476
## [2,] 0.9254157 2.1014262  0.1251530
## [3,] 1.0332958 0.8043002  0.7697818
##
## $orientation
##                 buy    social       educ
## buy     0.865903570 0.1944111  0.4608854
## social -0.008180761 0.9267628 -0.3755580
## educ   -0.500144062 0.3214267  0.8040776
```

```
#calculate number of variance parameters -12
nVarParams("VVE", d = 3, G = 3)
```

```
## [1] 12
```

Though the three clusters have the same descriptions, there is better distinction in the buying parameter. There isn't as much separation in the social feature for cluster 2 as in the previous model. – Cluster 1 - Non buyer – Cluster 2 - Non social – Cluster 3 - Ambitious

Mclust picked the VVE variance model. The class-conditional distributions are ellipsoidal; they allow for different variances in the feature space as well as correlation between features. They are ellipsoidal but equal in orientation.

There are 12 total estimated variance parameters.

d

I prefer the 2nd clustering model (VII) since it better allows for unequal cluster sizes compared to k means particularly for the cluster characterized as ambitious. In the VVE model, it is difficult to see the distinction between the non-social and ambitious cluster groupinging since the distribution for non-social allowed for much larger variances in the social metric. For instance, someone in the ambitious group may be classified to the non-social group if they really want to buy something.

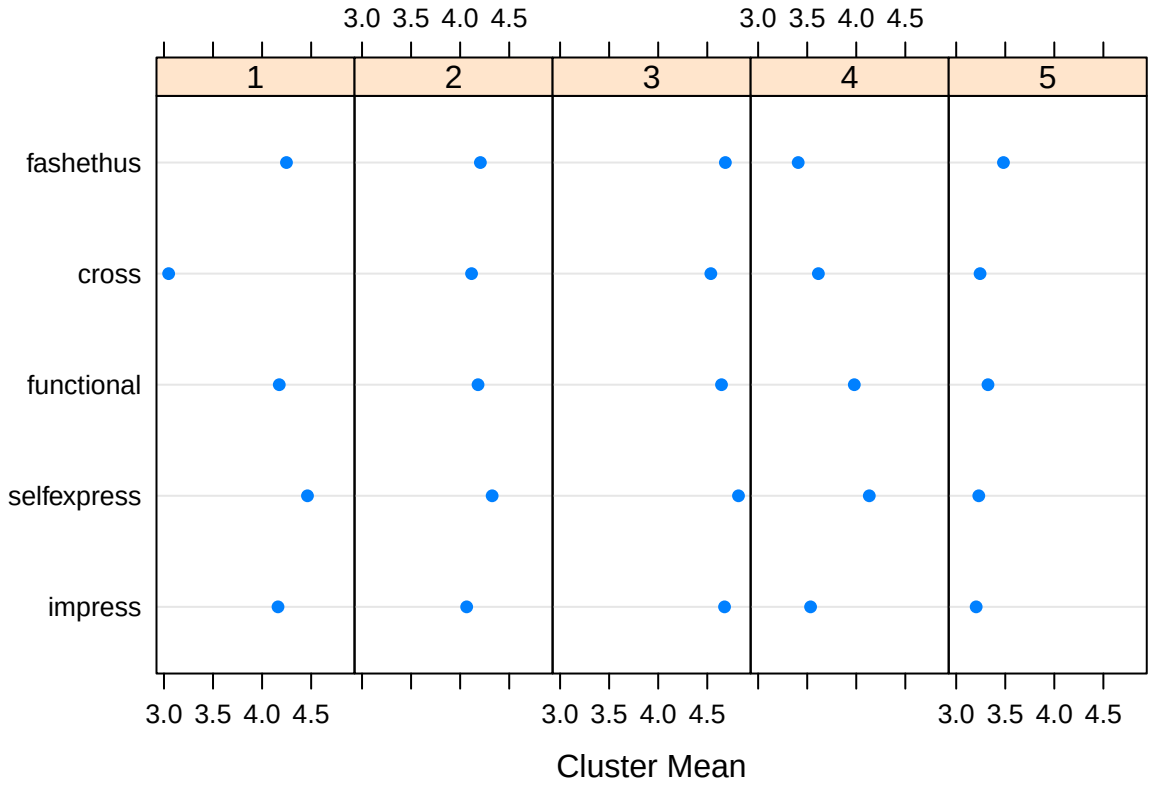
Problem 2

a

```
nuoqi <- read.csv('nuoqi.csv')
set.seed(12345)
fit = kmeans(nuoqi[,1:5], 5, 100, 100)
summary(fit)
```

```
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 147 0.15    4.16        4.46       4.17  3.05      4.25 0.4443
## 2 303 0.30    4.07        4.33       4.18  4.12      4.21 0.3704
## 3 240 0.24    4.68        4.82       4.65  4.54      4.68 0.3038
## 4 158 0.16    3.53        4.13       3.98  3.61      3.41 0.4500
## 5 146 0.15    3.20        3.23       3.32  3.24      3.48 0.5158
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4056
## SSE= 813.4389 ; SSB= 1156.095 ; SST= 1969.534
## R-Squared =  0.5869891
## Pseudo F =  351.4025
```

```
plot(fit)
```



There seems to be some overlap between the clusters with three distinct clusters (non enthusiasts/average/high enthusiasts) Cluster 1 - Non cross fashion Cluster 2 - scores average on all 5 sections Cluster 3 - scores high on all 5 sections Cluster 4 - functional and self expression Cluster 5 - scores low on all 5 sections - not bothered by fashion.

b

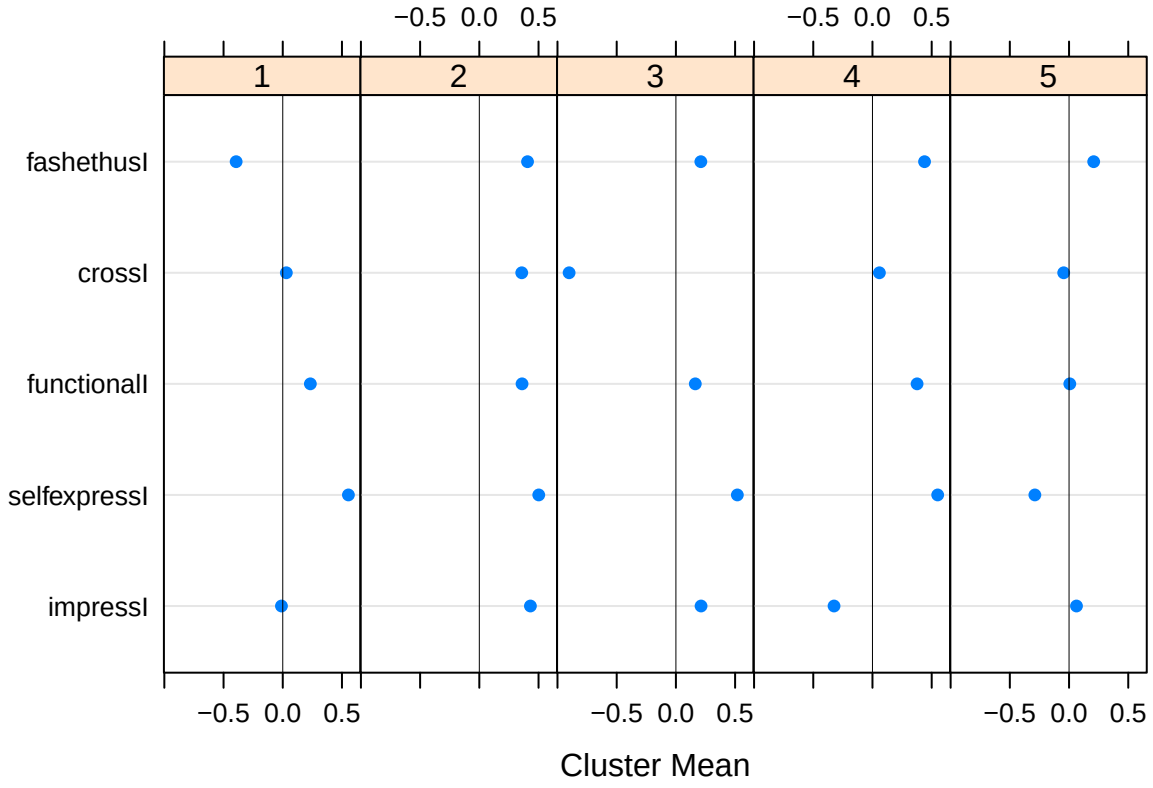
```
nuoqi$impressI = nuoqi$impress-nuoqi$xbar
nuoqi$selfexpressI = nuoqi$selfexpress-nuoqi$xbar
nuoqi$functionalI = nuoqi$functional-nuoqi$xbar
nuoqi$crossI = nuoqi$cross-nuoqi$xbar
nuoqi$fashethusI = nuoqi$fashethus-nuoqi$xbar
```

c

```
nuo1 <- nuoqi[c("impressI","selfexpressI","functionalI","crossI","fashethusI")]
set.seed(12345)
fit.nuoqiI = kmeans(nuo1, 5, nstart=100)

#Cluster sizes, means, RMSE
summary(fit.nuoqiI)
```

```
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 172 0.17    -0.01         0.55        0.23   0.03      -0.39 0.3720
## 2 310 0.31     0.43         0.50        0.36   0.36       0.41 0.3252
## 3 173 0.17     0.21         0.52        0.16  -0.90       0.21 0.4127
## 4 165 0.17    -0.32         0.55        0.38   0.06       0.44 0.3510
## 5 174 0.18     0.06        -0.29        0.01  -0.05       0.21 0.3988
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3672
## SSE= 666.8154 ; SSB= 444.9373 ; SST= 1111.753
## R-Squared =  0.4002125
## Pseudo F =  164.9793
```

```
plot(fit.nuoqiI)
```



Still overlapping clusters. Cluster 1 - Functional & SelfExpression Cluster 2 - Fashion conscious Cluster 3 - Non cross fashion Cluster 4 - Fashion enthusiast & self expression Cluster 5 - Fashion enthusiast as the driver.

Even with ipsatization, the clustering is not clear. In fact, R-square and pseudo-F both decrease.

d

```
# on original features
F = double(5)
SSE = double(5)
```

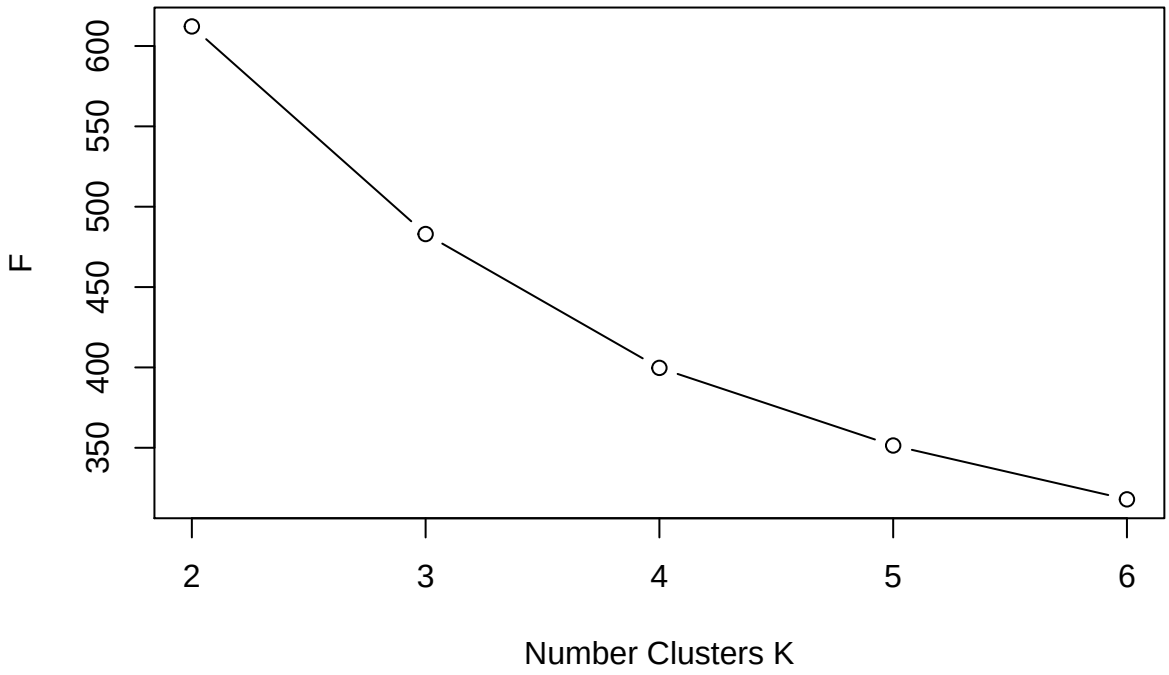
```
Rsqr = double(5)
for(K in 2:6){
  set.seed(12345)
  fit = kmeans(nuoqi[,1:5], K, nstart=100)
  F[K-1] = summary(fit)$F
  SSE[K-1] = fit$tot.withinss
  Rsqr[K-1] = summary(fit)$Rsqr
}
```

```
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 613 0.62    4.36        4.59       4.40  4.16      4.39 0.4416
## 2 381 0.38    3.47        3.76       3.71  3.36      3.61 0.5718
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4955
## SSE= 1217.932 ; SSB= 751.6017 ; SST= 1969.534
## R-Squared =  0.381614
## Pseudo F =  612.1761
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 613 0.62    4.36        4.59       4.40  4.16      4.39 0.4416
## 2 381 0.38    3.47        3.76       3.71  3.36      3.61 0.5718
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4955
## SSE= 1217.932 ; SSB= 751.6017 ; SST= 1969.534
## R-Squared =  0.381614
## Pseudo F =  612.1761
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 458 0.46    4.01        4.37       4.16  3.68      4.08 0.4606
## 2 234 0.24    3.28        3.48       3.53  3.33      3.46 0.5424
## 3 302 0.30    4.60        4.74       4.57  4.51      4.61 0.3359
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4486
## SSE= 997.3443 ; SSB= 972.1895 ; SST= 1969.534
## R-Squared =  0.493614
## Pseudo F =  483.0026
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 458 0.46    4.01        4.37       4.16  3.68      4.08 0.4606
## 2 234 0.24    3.28        3.48       3.53  3.33      3.46 0.5424
## 3 302 0.30    4.60        4.74       4.57  4.51      4.61 0.3359
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4486
## SSE= 997.3443 ; SSB= 972.1895 ; SST= 1969.534
## R-Squared =  0.493614
## Pseudo F =  483.0026
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 337 0.34    4.01        4.32       4.14  4.10      4.07 0.4040
## 2 211 0.21    3.25        3.42       3.51  3.34      3.44 0.5218
## 3 194 0.20    4.01        4.43       4.15  3.10      4.08 0.4671
## 4 252 0.25    4.67        4.80       4.63  4.53      4.68 0.3092
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4242
## SSE= 890.6949 ; SSB= 1078.839 ; SST= 1969.534
## R-Squared =  0.5477636
## Pseudo F =  399.7068
##
```

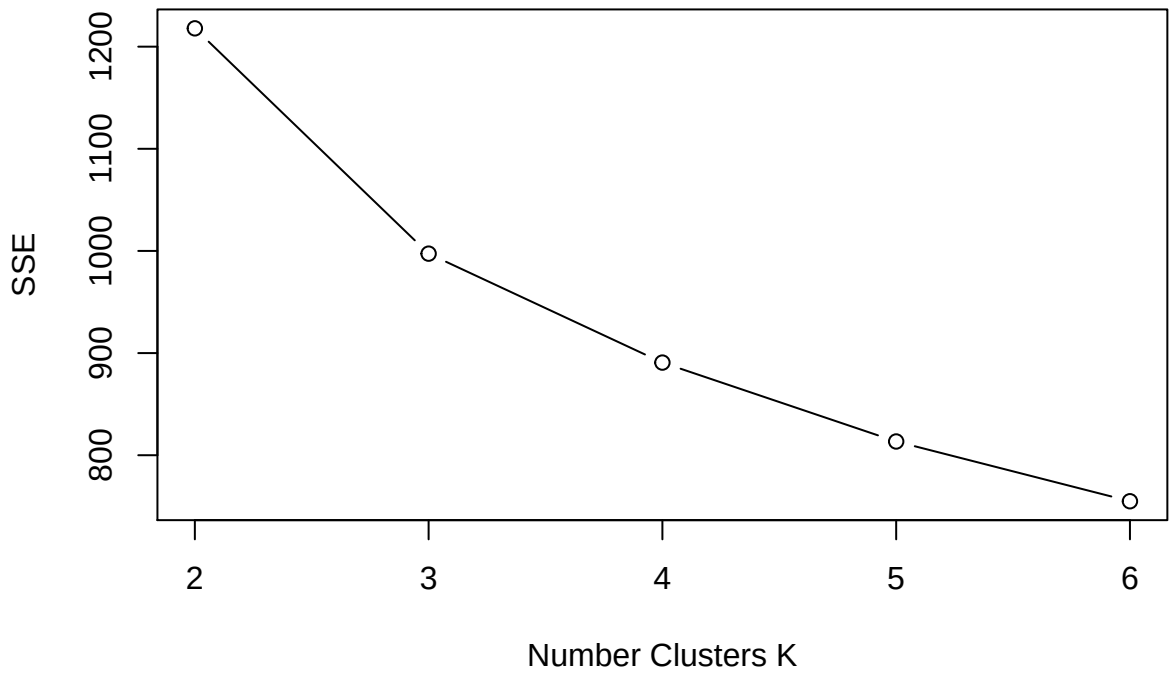
```
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 337 0.34    4.01        4.32       4.14  4.10      4.07 0.4040
## 2 211 0.21    3.25        3.42       3.51  3.34      3.44 0.5218
## 3 194 0.20    4.01        4.43       4.15  3.10      4.08 0.4671
## 4 252 0.25    4.67        4.80       4.63  4.53      4.68 0.3092
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4242
## SSE= 890.6949 ; SSB= 1078.839 ; SST= 1969.534
## R-Squared =  0.5477636
## Pseudo F =  399.7068
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 147 0.15    4.16        4.46       4.17  3.05      4.25 0.4443
## 2 303 0.30    4.07        4.33       4.18  4.12      4.21 0.3704
## 3 240 0.24    4.68        4.82       4.65  4.54      4.68 0.3038
## 4 158 0.16    3.53        4.13       3.98  3.61      3.41 0.4500
## 5 146 0.15    3.20        3.23       3.32  3.24      3.48 0.5158
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4056
## SSE= 813.4389 ; SSB= 1156.095 ; SST= 1969.534
## R-Squared =  0.5869891
## Pseudo F =  351.4025
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 147 0.15    4.16        4.46       4.17  3.05      4.25 0.4443
## 2 303 0.30    4.07        4.33       4.18  4.12      4.21 0.3704
## 3 240 0.24    4.68        4.82       4.65  4.54      4.68 0.3038
## 4 158 0.16    3.53        4.13       3.98  3.61      3.41 0.4500
## 5 146 0.15    3.20        3.23       3.32  3.24      3.48 0.5158
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4056
## SSE= 813.4389 ; SSB= 1156.095 ; SST= 1969.534
## R-Squared =  0.5869891
## Pseudo F =  351.4025
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 123 0.12    3.57        4.28       3.97  3.61      3.31 0.4301
## 2 106 0.11    3.48        3.33       3.79  3.76      3.90 0.4443
## 3  95 0.10    3.07        3.34       3.20  2.93      3.29 0.4946
## 4 231 0.23    4.69        4.82       4.66  4.55      4.70 0.2993
## 5 294 0.30    4.10        4.39       4.20  4.12      4.20 0.3521
## 6 145 0.15    4.17        4.47       4.18  3.06      4.26 0.4390
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.3909
## SSE= 754.938 ; SSB= 1214.596 ; SST= 1969.534
## R-Squared =  0.616692
## Pseudo F =  317.9124
##
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 123 0.12    3.57        4.28       3.97  3.61      3.31 0.4301
## 2 106 0.11    3.48        3.33       3.79  3.76      3.90 0.4443
## 3  95 0.10    3.07        3.34       3.20  2.93      3.29 0.4946
## 4 231 0.23    4.69        4.82       4.66  4.55      4.70 0.2993
## 5 294 0.30    4.10        4.39       4.20  4.12      4.20 0.3521
## 6 145 0.15    4.17        4.47       4.18  3.06      4.26 0.4390
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.3909
## SSE= 754.938 ; SSB= 1214.596 ; SST= 1969.534
## R-Squared =  0.616692
```

```
## Pseudo F =  317.9124
```

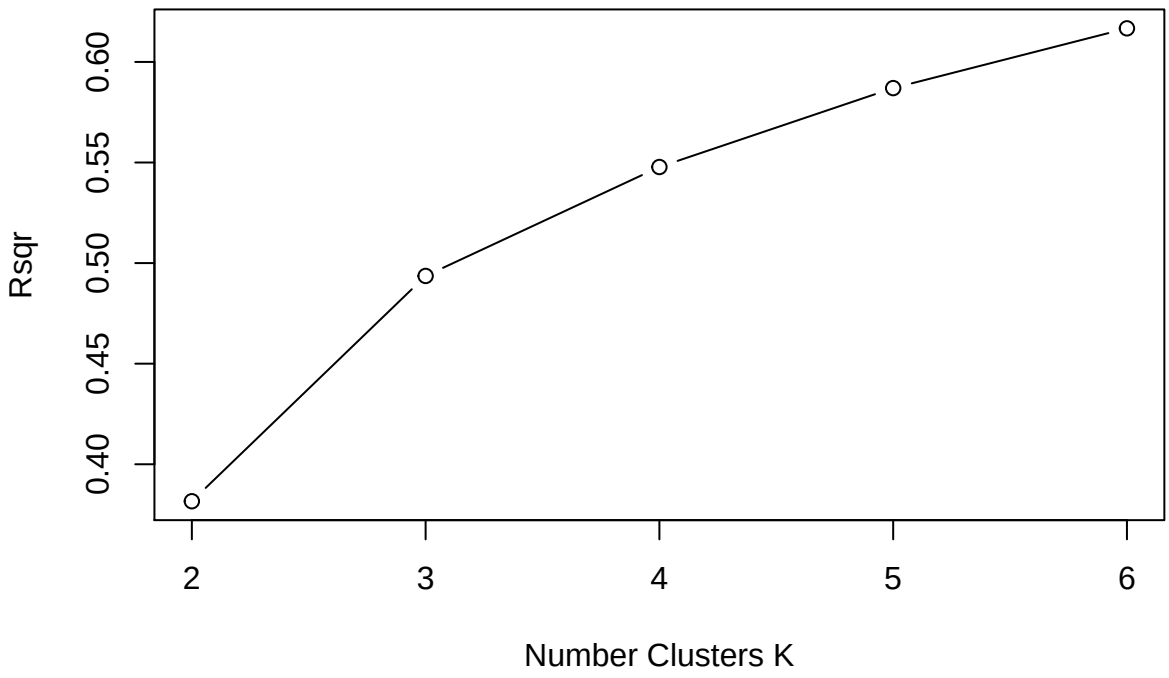
```
plot(2:6, F, type = "b",
    xlab = "Number Clusters K", ylab = 'F')
```




```
plot(2:6, SSE, type = "b",
    xlab = "Number Clusters K", ylab = 'SSE')
```




```
plot(2:6, Rsqr, type = "b",
    xlab = "Number Clusters K", ylab = 'Rsqr')
```




```
SSE
```

```
## [1] 1217.9321  997.3443  890.6949  813.4389  754.9380
```

```
F
```

```
## [1] 612.1761 483.0026 399.7068 351.4025 317.9124
```

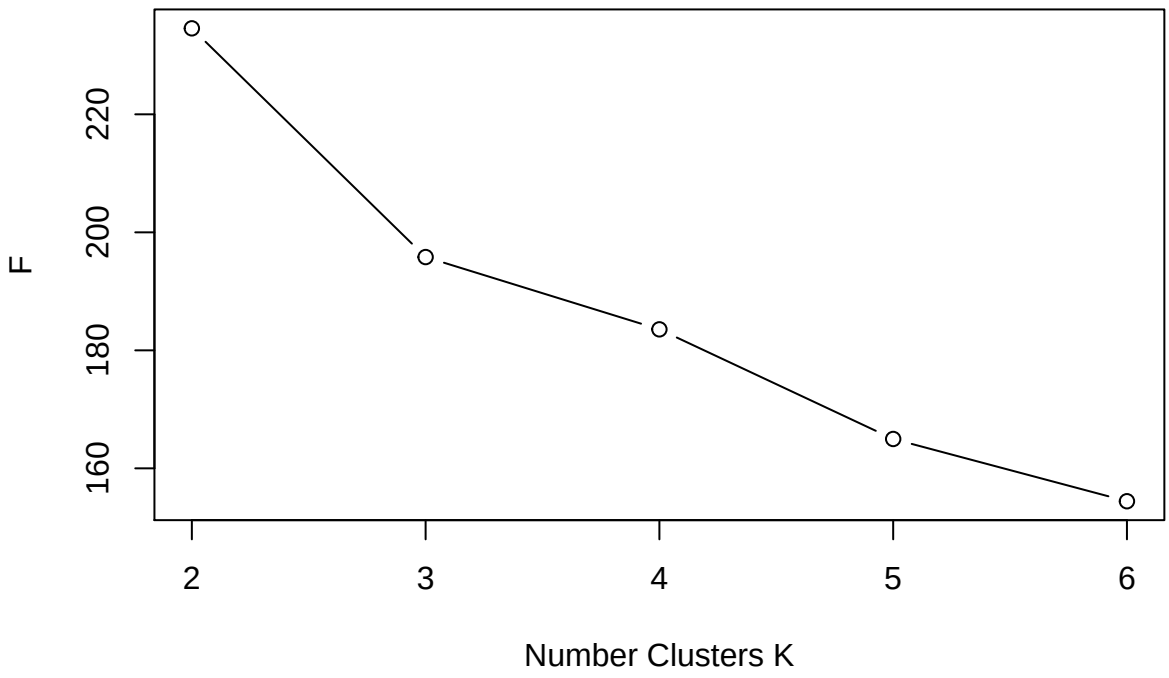
```
# on ipsatized feature
F_I = double(5)
SSE_I = double(5)
Rsqr_I = double(5)
for(K in 2:6){
  set.seed(12345)
  fit = kmeans(nuoqi[,(ncol(nuoqi)-4):ncol(nuoqi)], K, nstart=100)
  F_I[K-1] = summary(fit)$F
  SSE_I[K-1] = fit$tot.withinss
  Rsqr_I[K-1] = summary(fit)$Rsqr
}
```

```
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 620 0.62     0.19         0.38        0.29   0.30       0.28 0.3933
## 2 374 0.38     0.02         0.38        0.16  -0.60       0.07 0.4748
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.4258
```

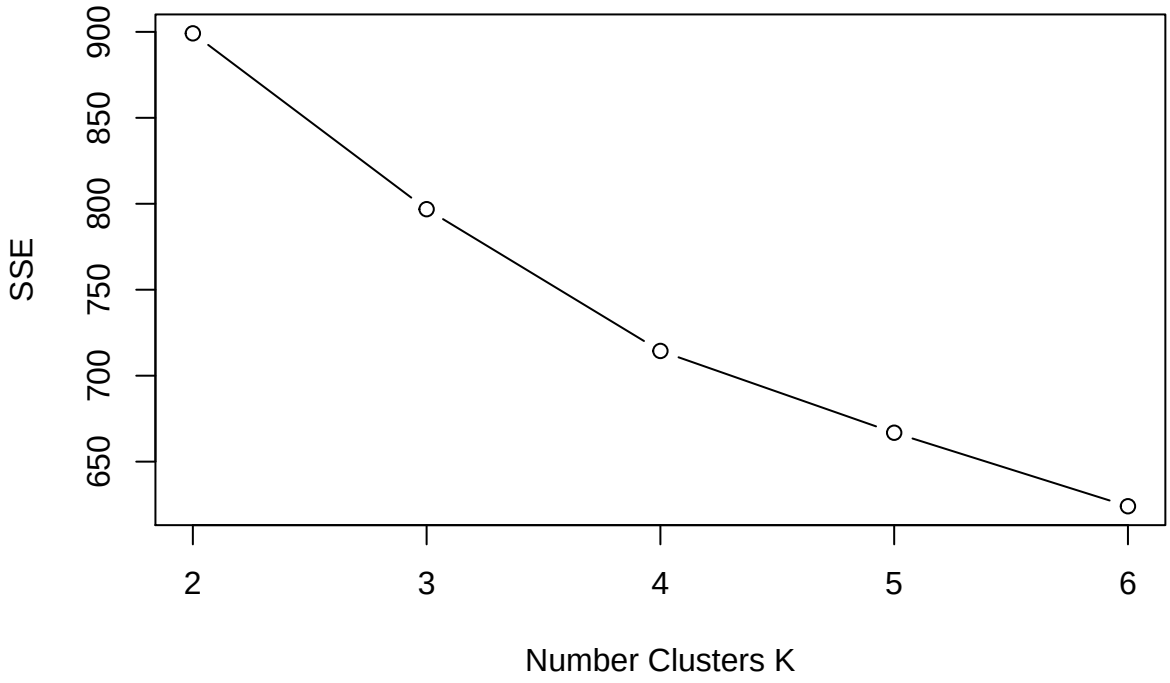
```
## SSE= 899.1399 ; SSB= 212.6127 ; SST= 1111.753
## R-Squared =  0.191241
## Pseudo F =  234.5707
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 620 0.62     0.19         0.38        0.29   0.30       0.28 0.3933
## 2 374 0.38     0.02         0.38        0.16  -0.60       0.07 0.4748
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.4258
## SSE= 899.1399 ; SSB= 212.6127 ; SST= 1111.753
## R-Squared =  0.191241
## Pseudo F =  234.5707
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 277 0.28     0.07         0.56        0.24  -0.70       0.07 0.4451
## 2 451 0.45     0.28         0.56        0.37   0.32       0.31 0.3557
## 3 266 0.27    -0.07        -0.10        0.03   0.04       0.17 0.4245
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.4010
## SSE= 796.8534 ; SSB= 314.8993 ; SST= 1111.753
## R-Squared =  0.2832458
## Pseudo F =  195.8109
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 277 0.28     0.07         0.56        0.24  -0.70       0.07 0.4451
## 2 451 0.45     0.28         0.56        0.37   0.32       0.31 0.3557
## 3 266 0.27    -0.07        -0.10        0.03   0.04       0.17 0.4245
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.4010
## SSE= 796.8534 ; SSB= 314.8993 ; SST= 1111.753
## R-Squared =  0.2832458
## Pseudo F =  195.8109
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 203 0.20     0.01        -0.25        0.05  -0.01       0.25 0.4098
## 2 189 0.19     0.21         0.54        0.17  -0.86       0.21 0.4168
## 3 231 0.23    -0.18         0.59        0.27   0.04      -0.21 0.3890
## 4 371 0.37     0.34         0.52        0.38   0.32       0.44 0.3349
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3799
## SSE= 714.3881 ; SSB= 397.3645 ; SST= 1111.753
## R-Squared =  0.3574217
## Pseudo F =  183.5561
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 203 0.20     0.01        -0.25        0.05  -0.01       0.25 0.4098
## 2 189 0.19     0.21         0.54        0.17  -0.86       0.21 0.4168
## 3 231 0.23    -0.18         0.59        0.27   0.04      -0.21 0.3890
## 4 371 0.37     0.34         0.52        0.38   0.32       0.44 0.3349
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3799
## SSE= 714.3881 ; SSB= 397.3645 ; SST= 1111.753
## R-Squared =  0.3574217
## Pseudo F =  183.5561
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 172 0.17    -0.01         0.55        0.23   0.03      -0.39 0.3720
## 2 310 0.31     0.43         0.50        0.36   0.36       0.41 0.3252
## 3 173 0.17     0.21         0.52        0.16  -0.90       0.21 0.4127
```

```
## 4 165 0.17    -0.32         0.55        0.38   0.06       0.44 0.3510
## 5 174 0.18     0.06        -0.29        0.01  -0.05       0.21 0.3988
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3672
## SSE= 666.8154 ; SSB= 444.9373 ; SST= 1111.753
## R-Squared =  0.4002125
## Pseudo F =  164.9793
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 172 0.17    -0.01         0.55        0.23   0.03      -0.39 0.3720
## 2 310 0.31     0.43         0.50        0.36   0.36       0.41 0.3252
## 3 173 0.17     0.21         0.52        0.16  -0.90       0.21 0.4127
## 4 165 0.17    -0.32         0.55        0.38   0.06       0.44 0.3510
## 5 174 0.18     0.06        -0.29        0.01  -0.05       0.21 0.3988
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3672
## SSE= 666.8154 ; SSB= 444.9373 ; SST= 1111.753
## R-Squared =  0.4002125
## Pseudo F =  164.9793
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 122 0.12     0.36         0.57        0.73   0.57       0.58 0.3744
## 2 139 0.14    -0.04        -0.35       -0.07  -0.09       0.20 0.4213
## 3 321 0.32     0.33         0.38        0.16   0.19       0.26 0.2758
## 4 133 0.13    -0.38         0.70        0.25  -0.13       0.41 0.3457
## 5 136 0.14    -0.03         0.57        0.39   0.01      -0.47 0.3712
## 6 143 0.14     0.26         0.47        0.17  -0.98       0.21 0.4165
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3554
## SSE= 624.0503 ; SSB= 487.7024 ; SST= 1111.753
## R-Squared =  0.4386788
## Pseudo F =  154.4266
##
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 122 0.12     0.36         0.57        0.73   0.57       0.58 0.3744
## 2 139 0.14    -0.04        -0.35       -0.07  -0.09       0.20 0.4213
## 3 321 0.32     0.33         0.38        0.16   0.19       0.26 0.2758
## 4 133 0.13    -0.38         0.70        0.25  -0.13       0.41 0.3457
## 5 136 0.14    -0.03         0.57        0.39   0.01      -0.47 0.3712
## 6 143 0.14     0.26         0.47        0.17  -0.98       0.21 0.4165
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3554
## SSE= 624.0503 ; SSB= 487.7024 ; SST= 1111.753
## R-Squared =  0.4386788
## Pseudo F =  154.4266
```

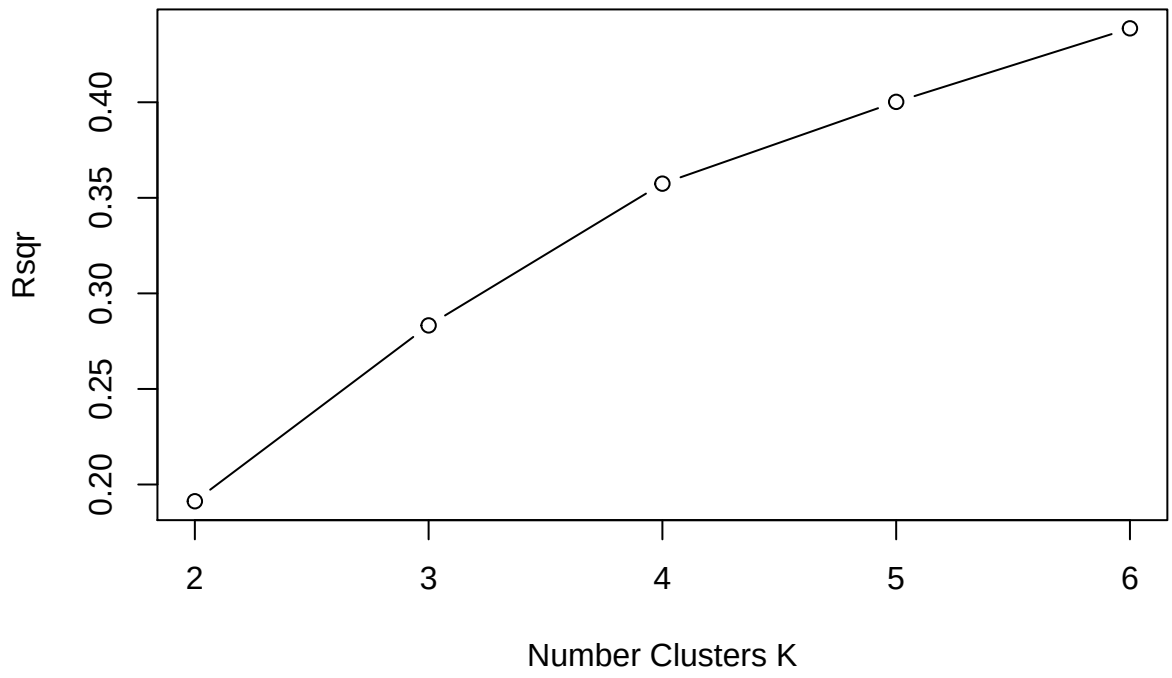
```
plot(2:6, F_I, type = "b",
    xlab = "Number Clusters K", ylab = 'F')
```




```
plot(2:6, SSE_I, type = "b",
    xlab = "Number Clusters K", ylab = 'SSE')
```




```
plot(2:6, Rsqr_I, type = "b",
    xlab = "Number Clusters K", ylab = 'Rsqr')
```




```
SSE_I
```

```
## [1] 899.1399 796.8534 714.3881 666.8154 624.0503
```

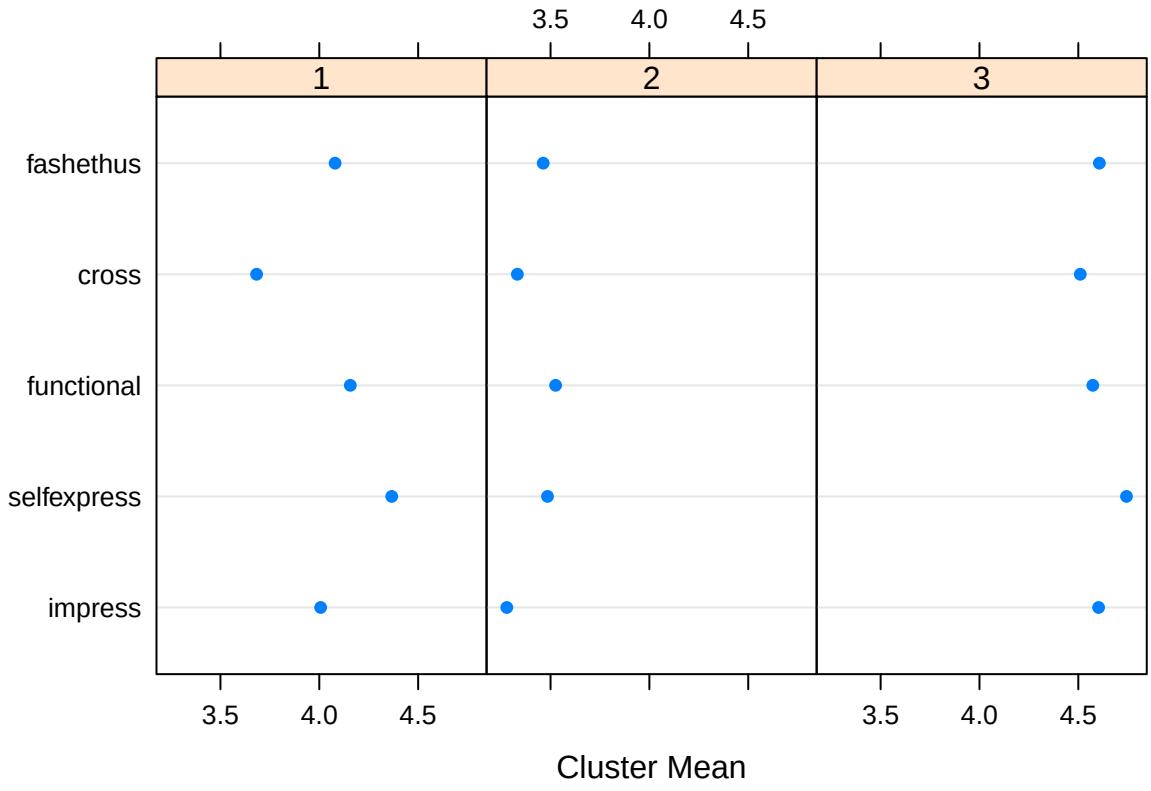
```
F_I
```

```
## [1] 234.5707 195.8109 183.5561 164.9793 154.4266
```

```
set.seed(12345)

#SSE elbow at 3 for orig data
fit_13 = kmeans(nuoqi[,1:5], 3, nstart=100)
summary(fit_13)
```

```
##     n  Pct impress selfexpress functional cross fashethus   RMSE
## 1 458 0.46    4.01        4.37       4.16  3.68      4.08 0.4606
## 2 234 0.24    3.28        3.48       3.53  3.33      3.46 0.5424
## 3 302 0.30    4.60        4.74       4.57  4.51      4.61 0.3359
##   994 1.00    4.02        4.27       4.13  3.85      4.09 0.4486
## SSE= 997.3443 ; SSB= 972.1895 ; SST= 1969.534
## R-Squared =  0.493614
## Pseudo F =  483.0026
```

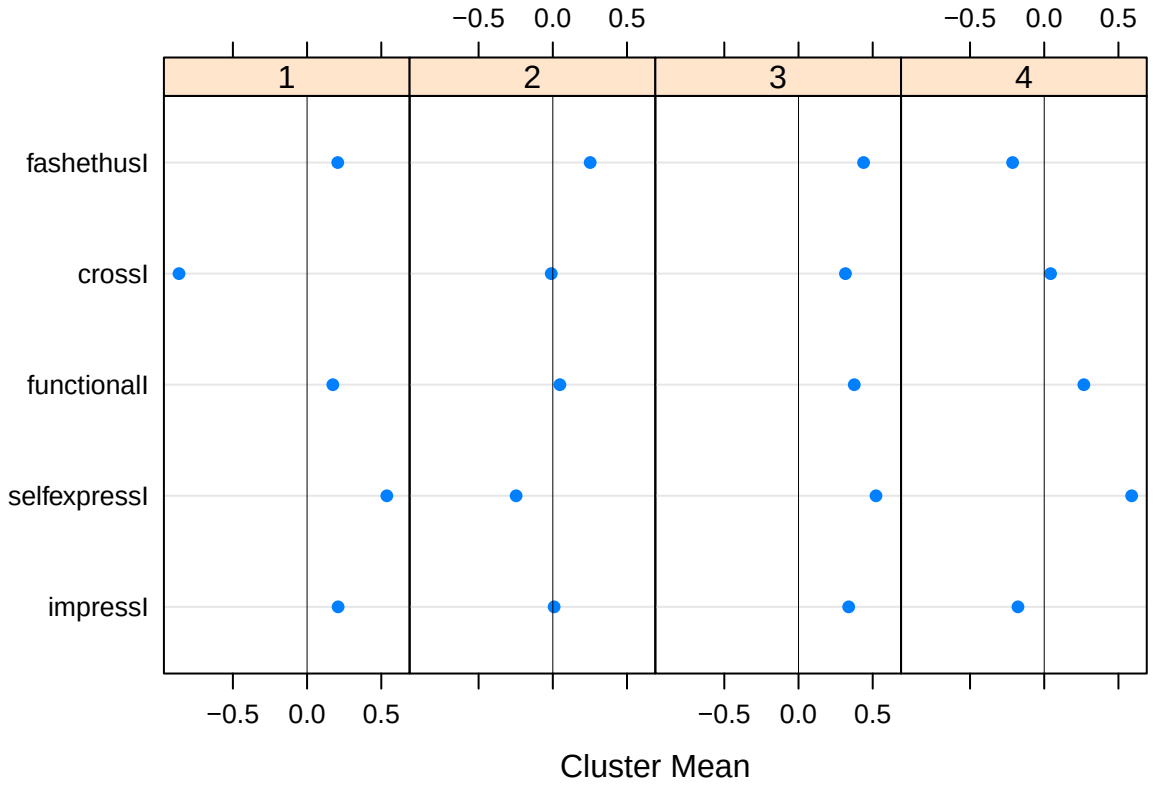
```r
plot(fit_13)
```




```r
#SSE elbow at K = 4 for ipsatized data
fit_24 = kmeans(nuoqi[,(ncol(nuoqi)-4):ncol(nuoqi)], 4, nstart=100)
summary(fit_24)
```

```
##     n  Pct impressI selfexpressI functionalI crossI fashethusI   RMSE
## 1 189 0.19     0.21         0.54        0.17  -0.86       0.21 0.4168
## 2 203 0.20     0.01        -0.25        0.05  -0.01       0.25 0.4098
## 3 371 0.37     0.34         0.52        0.38   0.32       0.44 0.3349
## 4 231 0.23    -0.18         0.59        0.27   0.04      -0.21 0.3890
##   994 1.00     0.13         0.38        0.24  -0.04       0.20 0.3799
## SSE= 714.3881 ; SSB= 397.3645 ; SST= 1111.753
## R-Squared =  0.3574217
## Pseudo F =  183.5561
```

```r
plot(fit_24)
```



3 clusters with raw data seems to provide the most distinctly characterized clusters, but this may be clustering on the underlying attitudes of shoppers (i.e. there are 2 distinct clusters who tend to respond more positively and another that tend to respond negatively).

The ipsatized data shows that 4 clusters may be ideal, as it gives 4 distinct profiles. Cluster 1 is neutral across all dimensions. Cluster 2 is heavy on self expression, cluster 3 against cross fashion, and cluster 4 who is enthusiastic across the board. These clusters may be more actionable and thus I would recommend this.

e

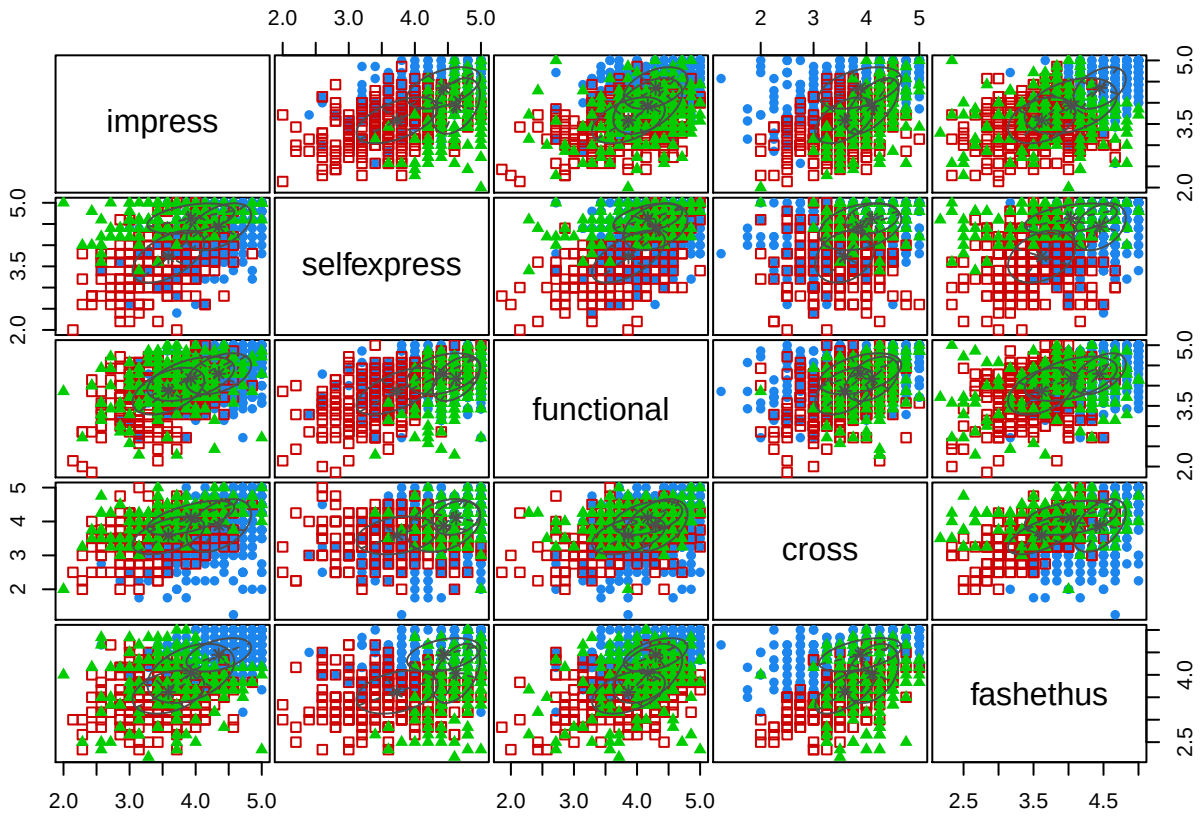
```
fit.nuo.gmm = Mclust(nuoqi[,1:5],G=3)
fit.nuo.gmm$parameters$mean
```

```
##                 [,1]     [,2]     [,3]
## impress     4.350528 3.587571 3.926259
## selfexpress 4.435210 3.730724 4.626003
## functional  4.294184 3.864028 4.169933
## cross       3.882437 3.588585 4.107619
## fashethus   4.445661 3.615950 4.033125
```

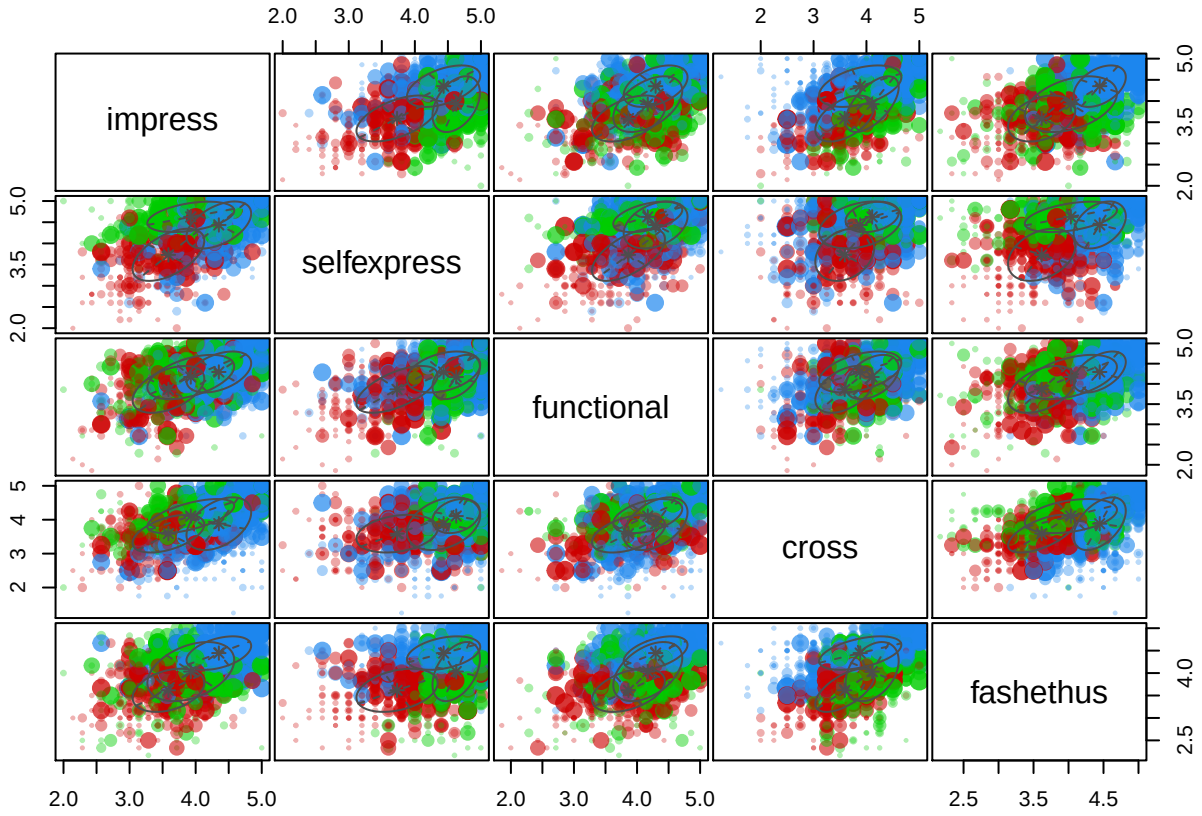
```
fit.nuo.gmm$parameters$pro
```

```
## [1] 0.4504905 0.2977975 0.2517120
```

```
plot(fit.nuo.gmm, what = "classification")
```




```
plot(fit.nuo.gmm, what = "uncertainty")
```



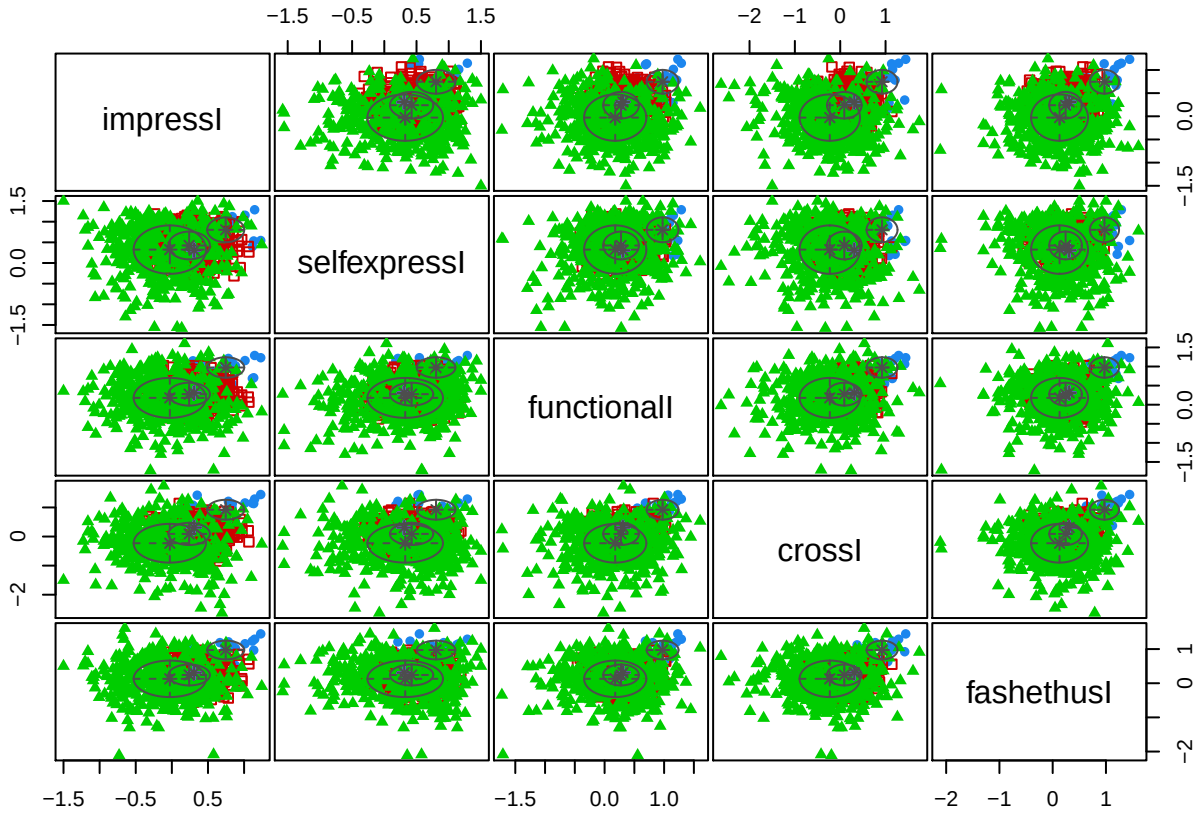

```
fit.nuo.gmm1 = Mclust(nuoqi[,(ncol(nuoqi)-4):ncol(nuoqi)],G=4)
fit.nuo.gmm1$parameters$mean
```

```
##                    [,1]       [,2]        [,3]      [,4]
## impressI      0.7454332 0.23948700 -0.02827033 0.3087305
## selfexpressI 0.8034411 0.42382292  0.32428608 0.3087305
## functionalI   0.9743921 0.27601032  0.17856040 0.3087305
## crossI        0.9224388 0.09042632 -0.22986097 0.3087305
## fashethusI    0.9717990 0.24199952  0.12905789 0.3087305
```

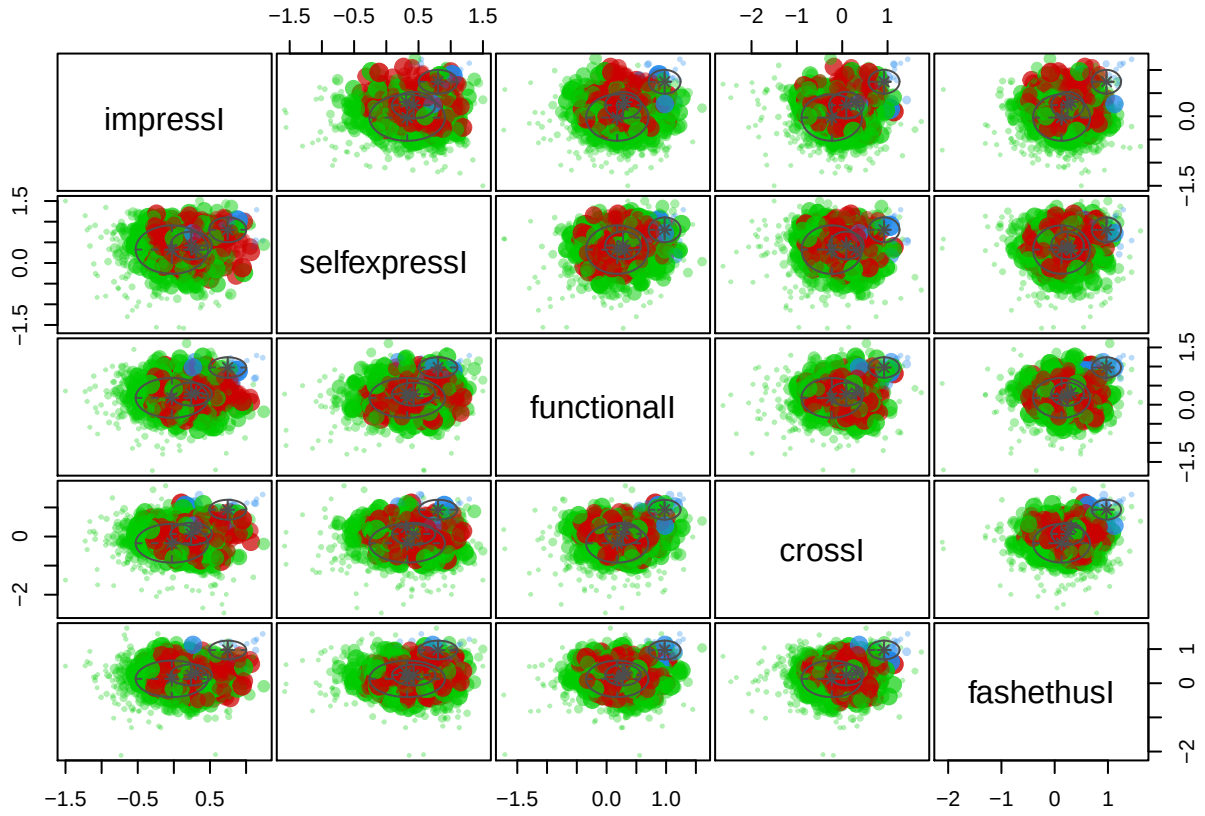
```
fit.nuo.gmm1$parameters$pro
```

```
## [1] 0.01925018 0.50533667 0.46132914 0.01408400
```

```
plot(fit.nuo.gmm1, what = "classification")
```




```
plot(fit.nuo.gmm1, what = "uncertainty")
```



When using the Gausian mixture models, the uncertainty is really high as the clustering distributions overlap a lot and there is minimal distinction between responses.

Problem 3

```
mulist = c(0.5,1,2)
n = 3000

for (i in c(1,2,3)) {
  set.seed(421)
  mu = mulist[i]
  data = c(rnorm(n, mean=(0-mu),sd=1), rnorm(n,mean=mu,sd=1))
  result = data.frame(mean=c(0,0),variance=c(0,0),row.names = c("cluster1","cluster2"))

  # k-means
  set.seed(100)
  fit1 = kmeans(data, 2, 100, 100)
  #fit1$centers
  df1 = data.frame(data=data,cluster=fit1$cluster)
  result[,1] = fit1$centers
  result[1,2]=var(subset(df1,df1$cluster==1)$data)
  result[2,2]=var(subset(df1,df1$cluster==2)$data)
  print(result)
```

```
  # GMM
  set.seed(100)
  fit2 = Mclust(data, G=2)
  fit2$parameter$variance$sigmasq
  fit2$parameter$mean
  df2 = data.frame(data=data,cluster=fit2[["classification"]])
  result[1,1] = fit2[["parameters"]][["mean"]][["2"]]
  result[2,1] = fit2[["parameters"]][["mean"]][["1"]]
  result[1,2]=var(subset(df2,df2$cluster==1)$data)
  result[2,2]=var(subset(df2,df2$cluster==2)$data)
  result$sigmasq = fit2$parameters$variance$sigmasq
  print(result)

}
```

```
##                mean  variance
## cluster1  0.8907177 0.4562486
## cluster2 -0.8807468 0.4412851
##                mean  variance   sigmasq
## cluster1  0.5540589 0.4504137 0.9416563
## cluster2 -0.5258721 0.4476960 0.9416563
##               mean  variance
## cluster1  1.189412 0.6406970
## cluster2 -1.134693 0.6295756
##                mean  variance   sigmasq
## cluster1  1.0278613 0.6381206 0.9902212
## cluster2 -0.9672354 0.6320593 0.9902212
##               mean  variance
## cluster1  2.040625 0.9197624
## cluster2 -1.995087 0.9136479
##               mean  variance   sigmasq
## cluster1  2.023800 0.9136479 0.9799661
## cluster2 -1.980256 0.9197624 0.9799661
```

K-means will make the variance of each cluster smaller than before, and the mean will be pulled to the 2 sides as it is biased towards separating the means. However, as mu gets larger, the effect gets smaller as the two populations get far apart and both K-means and GMM can deal with them. GMM can deal with smaller mu better than k-means.