Clustering Similarity

Code

In this example, clustering algorithms are performed on a dataset of automobile measurements for acceleration and rotation The goal is to separate the data into classes of driving It is known that there are three classes (NORMAL, SLOW, AGGRESSIVE), but this can still be explored

kMeans Setup

Hide

```
# Read in train and test sets
dataTrain <- read.csv(".//train_motion_data.csv")
dataTest <- read.csv(".//test_motion_data.csv")

dataClasses <- dataTrain[7]

# Scale sets
dataTrain <- scale(dataTrain[-7])
dataTest <- scale(dataTest[-7])

# Concat class column
dataTrain <- cbind(dataTrain, dataClasses)
```

kMeans Clustering

Hide

```
dataClusterKMeans <- kmeans(dataTrain[, 1:7], 3, nstart=20)
dataClusterKMeans
```

```
K-means clustering with 3 clusters of sizes 1007, 1333, 1304

Cluster means:
        AccX        AccY         AccZ      GyroX       GyroY       GyroZ  Timestamp
1  0.8734311  0.58509336 -0.004270451 -0.4832429 -0.21583377 -0.56458585 -0.3905976
2 -0.1163214  0.01349413 -0.027434929  0.1193752  0.06102417  0.04635593  1.0173109
3 -0.5555896 -0.46562630  0.031342872  0.2511491  0.10429402  0.38860851 -0.7383004

Clustering vector:
   [1] 3 3 3 3 3 3 3 3 1 3 1 1 3 3 1 1 1 1 1 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 1 3 3 3 3 3 3 1 3
1 1 1 1 3 3 3 3 1 3 3 3 3 1 3 3 3 3 3 3 3 3
  [76] 1 3 3 1 1 1 3 1 3 1 1 1 1 1 1 1 1 1 3 3 3 3 1 3 3 3 3 3 1 3 3 3 3 1 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 3 1
3 3 3 3 3 3 3 3 3 1 3 3 3 3 1 3 3 3 3 3 3 1
 [151] 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 1 1 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 1 3 1 1 1 3 3 3 3 3 3 3 3
3 3 3 3 3 1 3 1 1 1 3 1 3 3 3 3 1 1 1 1 3 3
 [226] 3 3 3 3 1 1 3 1 3 3 3 1 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 3 3 3 3 1 3 3 3 1 3 3 3 3 3 3 1 3 3
3 1 3 1 1 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 1
 [301] 3 3 3 3 3 3 3 3 3 1 3 3 1 1 3 3 3 1 3 3 3 1 3 3 1 1 3 3 3 3 1 3 3 3 3 3 1 1 3 3 3 3 3 3 3 1 3 3 3 3 3 1 1 1
3 3 3 3 3 3 3 3 1 1 3 1 3 3 3 3 1 3 1 3 1 3
 [376] 1 3 3 3 3 1 3 3 3 3 3 1 3 1 3 1 3 1 1 3 1 1 1 3 1 1 3 1 3 1 1 1 3 3 3 3 3 1 3 3 1 3 3 3 3 3 1 3 1 3 3 1 3
3 3 3 1 3 1 3 1 1 1 1 3 1 3 3 3 3 3 3 3 3 3
 [451] 3 3 3 3 3 1 3 1 3 3 1 1 1 3 3 3 3 1 3 3 1 1 1 3 3 1 3 3 1 3 3 1 1 3 3 3 1 3 3 3 3 3 1 3 3 3 1 1 1 3 3 3
1 1 1 3 1 3 3 1 1 3 3 3 1 3 3 3 3 3 3 1 3 1
 [526] 1 3 1 1 1 3 1 3 1 3 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3
1 1 3 1 1 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3
 [601] 3 1 1 1 1 1 1 3 3 3 1 3 3 1 3 3 3 3 1 3 3 3 3 3 3 3 1 1 3 3 3 3 3 1 1 1 3 3 3 1 3 1 3 3 3 1 3 3 3 3 3 3 3
3 1 1 1 3 1 1 1 1 1 3 1 1 1 1 1 1 3 1 3 1 1 3 3
 [676] 3 3 1 1 1 3 1 3 1 3 1 3 1 1 3 3 3 3 3 1 3 1 1 3 3 3 3 1 3 1 1 1 3 1 1 3 1 1 3 3 1 3 3 3 3 1 3 3 1 1 3 3 1
3 1 1 3 1 3 3 3 1 3 1 3 3 1 1 3 1 3 1 3 1 1
 [751] 3 1 3 3 3 3 3 3 1 1 1 1 1 1 1 3 3 3 1 1 1 1 1 1 3 1 1 1 1 3 1 3 3 1 3 1 3 1 3 3 1 3 3 1 3 3 3 1 3 1 1 1 1
1 1 1 1 3 3 1 3 1 1 3 1 3 3 1 1 1 1 1 3 1 1
 [826] 3 3 3 3 3 3 1 3 3 3 1 3 3 3 3 3 3 1 3 1 3 3 1 1 1 1 3 3 1 1 3 1 1 3 3 1 3 1 1 1 1 1 1 1 1 1 3 1 3 3 3 3 3 1
3 3 1 3 3 3 3 1 3 1 1 1 3 3 3 1 1 1 1 1
 [901] 1 3 1 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 1 3 1 3 1 3 1 3 1 3 1 3 3 3 3 3 3 1 1 3 1 1 1 3 1
1 1 1 3 1 3 1 3 1 1 3 3 1 1 1 1 1 3 3 1 3 3
 [976] 3 3 3 1 3 1 3 3 3 3 1 1 3 1 1 1 1 1 1 3 1 1 3 3 3
 [ reached getOption("max.print") -- omitted 2644 entries ]

Within cluster sum of squares by cluster:
[1] 7434.685 5514.851 7565.473
 (between_SS / total_SS =  19.6 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "ite
r"         "ifault"
```

Comparing Clusters with Classes

Hide

```
table(dataClusterKMeans$cluster, dataTrain$Class)
```

```
   
    AGGRESSIVE NORMAL SLOW
  1        414    452  141
  2        165      0 1168
  3        534    748   22
```

Heirarchical Clustering

Hide

```
# Subset the training data
set.seed(1324)
i <- sample(1:nrow(dataTrain), 50, replace=FALSE)

hdataTrain <- dataTrain[i,]

# Display Counts of Class
nrow(hdataTrain[hdataTrain$Class=="AGGRESSIVE",])
```

```
[1] 12
```

Hide

```
nrow(hdataTrain[hdataTrain$Class=="NORMAL",])
```

```
[1] 19
```

Hide

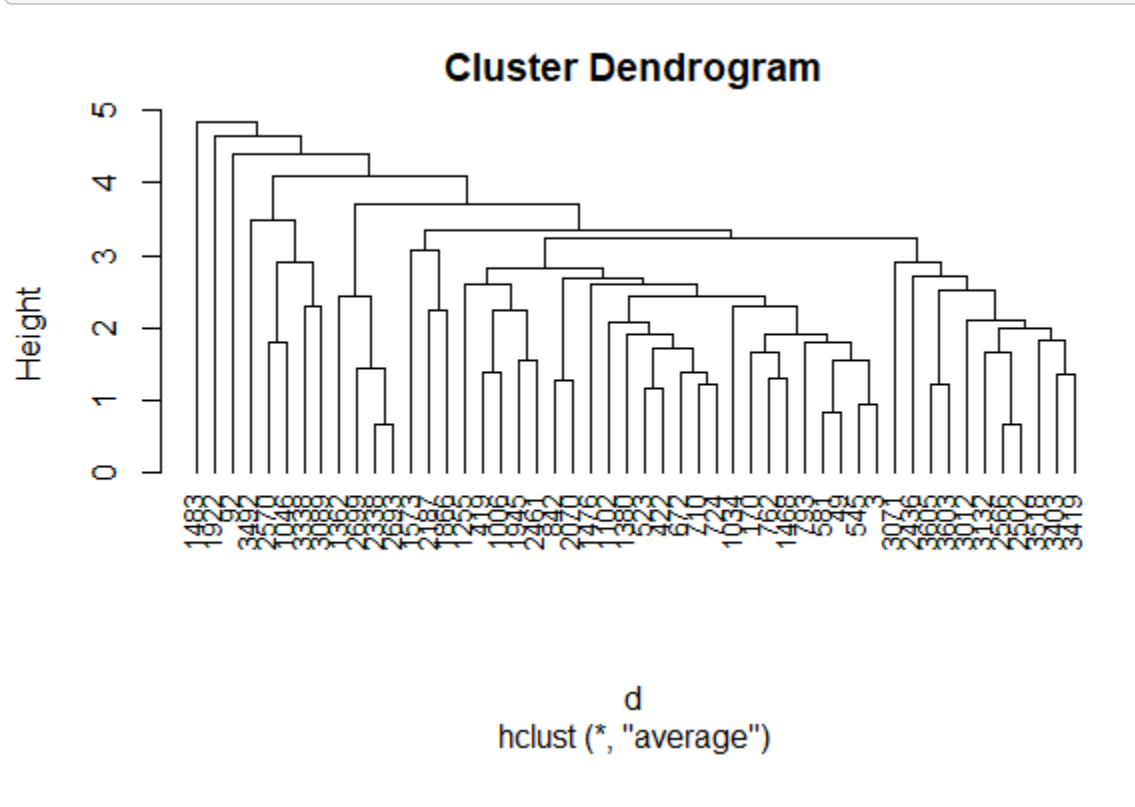
```
nrow(hdataTrain[hdataTrain$Class=="SLOW",])
```

```
[1] 19
```

Adjust distances

Hide

```
d <- dist(hdataTrain[-8])
fit.average <- hclust(d, method="average")
plot(fit.average, hang=-1, cex=.8)
```



Cutting dendogram

Hide

```
for(c in 1:9){
  cluster_cut <- cutree(fit.average, c)
  table_cut <- table(cluster_cut, hdataTrain$Class)
  print(table_cut)
  #ri <- rand.index(table_cut)
  print(paste("cut=", c))
}
```

```
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1         12     19   19
[1] "cut= 1"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1         11     19   19
          2          1      0    0
[1] "cut= 2"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1         10     19   19
          2          1      0    0
          3          1      0    0
[1] "cut= 3"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1         10     18   19
          2          0      1    0
          3          1      0    0
          4          1      0    0
[1] "cut= 4"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1         10     17   15
          2          0      1    4
          3          0      1    0
          4          1      0    0
          5          1      0    0
[1] "cut= 5"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1          9     17   12
          2          0      1    4
          3          1      0    3
          4          0      1    0
          5          1      0    0
          6          1      0    0
[1] "cut= 6"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1          9     17   12
          2          0      1    3
          3          1      0    3
          4          0      1    0
          5          1      0    0
          6          1      0    0
          7          0      0    1
[1] "cut= 7"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1          6     17   12
          2          3      0    0
          3          0      1    3
          4          1      0    3
          5          0      1    0
          6          1      0    0
          7          1      0    0
          8          0      0    1
[1] "cut= 8"
           
cluster_cut AGGRESSIVE NORMAL SLOW
          1          6     17    1
          2          0      0   11
          3          3      0    0
          4          0      1    3
          5          1      0    3
          6          0      1    0
          7          1      0    0
          8          1      0    0
          9          0      0    1
[1] "cut= 9"
```

As can be seen above, hierarchical clustering doesn't work well with this dataset since there isn't hierarchy in the different driving classes

Model Based Clustering

Hide

```
library(mclust)
fit <- Mclust(dataTrain[-8])
```

```
fitting ...

  |                                                             |   0%
  |                                                             |   1%
  |=                                                            |   2%
  |==                                                           |   3%
  |==                                                           |   4%
  |===                                                          |   5%
  |===                                                          |   6%
  |====                                                         |   6%
  |====                                                         |   7%
  |=====                                                        |   8%
  |=====                                                        |   9%
  |======                                                       |   9%
  |======                                                       |  10%
  |=======                                                      |  11%
  |=======                                                      |  12%
  |========                                                     |  13%
  |=========                                                    |  14%
  |=========                                                    |  15%
  |==========                                                   |  16%
  |==========                                                   |  17%
  |===========                                                  |  17%
  |===========                                                  |  18%
  |============                                                 |  19%
  |============                                                 |  20%
  |=============                                                |  21%
  |=============                                                |  22%
  |==============                                               |  23%
  |==============                                               |  24%
  |===============                                              |  24%
  |===============                                              |  25%
  |================                                             |  26%
  |================                                             |  27%
  |=================                                            |  28%
  |==================                                           |  29%
  |==================                                           |  30%
  |===================                                          |  31%
  |====================                                         |  32%
  |====================                                         |  33%
  |=====================                                        |  34%
  |=====================                                        |  35%
  |======================                                       |  35%
  |======================                                       |  36%
  |=======================                                      |  37%
  |=======================                                      |  38%
  |========================                                     |  39%
  |========================                                     |  40%
  |=========================                                    |  41%
  |=========================                                    |  42%
  |==========================                                   |  43%
  |===========================                                  |  44%
  |===========================                                  |  45%
  |============================                                 |  46%
  |=============================                                |  47%
  |=============================                                |  48%
  |==============================                               |  49%
  |==============================                               |  50%
  |===============================                              |  50%
  |===============================                              |  51%
  |================================                             |  52%
  |================================                             |  53%
  |=================================                            |  54%
  |==================================                           |  55%
  |==================================                           |  56%
  |===================================                          |  57%
  |====================================                         |  58%
  |====================================                         |  59%
  |=====================================                        |  60%
  |=====================================                        |  61%
  |======================================                       |  62%
  |======================================                       |  63%
  |=======================================                      |  64%
  |=======================================                      |  65%
  |========================================                     |  65%
  |========================================                     |  66%
  |=========================================                    |  67%
  |=========================================                    |  68%
  |==========================================                   |  69%
  |===========================================                  |  70%
  |===========================================                  |  71%
  |============================================                 |  72%
  |=============================================                |  73%
  |=============================================                |  74%
  |==============================================               |  75%
  |==============================================               |  76%
  |===============================================              |  76%
  |===============================================              |  77%
  |================================================             |  78%
  |================================================             |  79%
  |=================================================            |  80%
  |=================================================            |  81%
  |==================================================           |  82%
  |==================================================           |  83%
  |===================================================          |  83%
  |===================================================          |  84%
  |====================================================         |  85%
  |====================================================         |  86%
  |=====================================================        |  87%
  |======================================================       |  88%
  |======================================================       |  89%
  |=======================================================      |  90%
  |=======================================================      |  91%
  |========================================================     |  91%
  |========================================================     |  92%
  |=========================================================    |  93%
  |=========================================================    |  94%
  |==========================================================   |  94%
  |==========================================================   |  95%
  |===========================================================  |  96%
  |===========================================================  |  97%
  |============================================================ |  98%
  |=============================================================|  99%
  |=============================================================| 100%
```

Hide

```
summary(fit)
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VVE (ellipsoidal, equal orientation) model with 8 components:
```

	log-likelihood	n	df	BIC	ICL
	<dbl>	<int>	<dbl>	<dbl>	<dbl>
	-31527.45	3644	140	-64203.01	-65756.69

1 row

```
Clustering table:
  1   2   3   4   5   6   7   8 
997 245 486 810 624 134  96 252 
```

Hide

```
plot(fit)
```

```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density

Enter an item from the menu, or 0 to exit
```

Hide

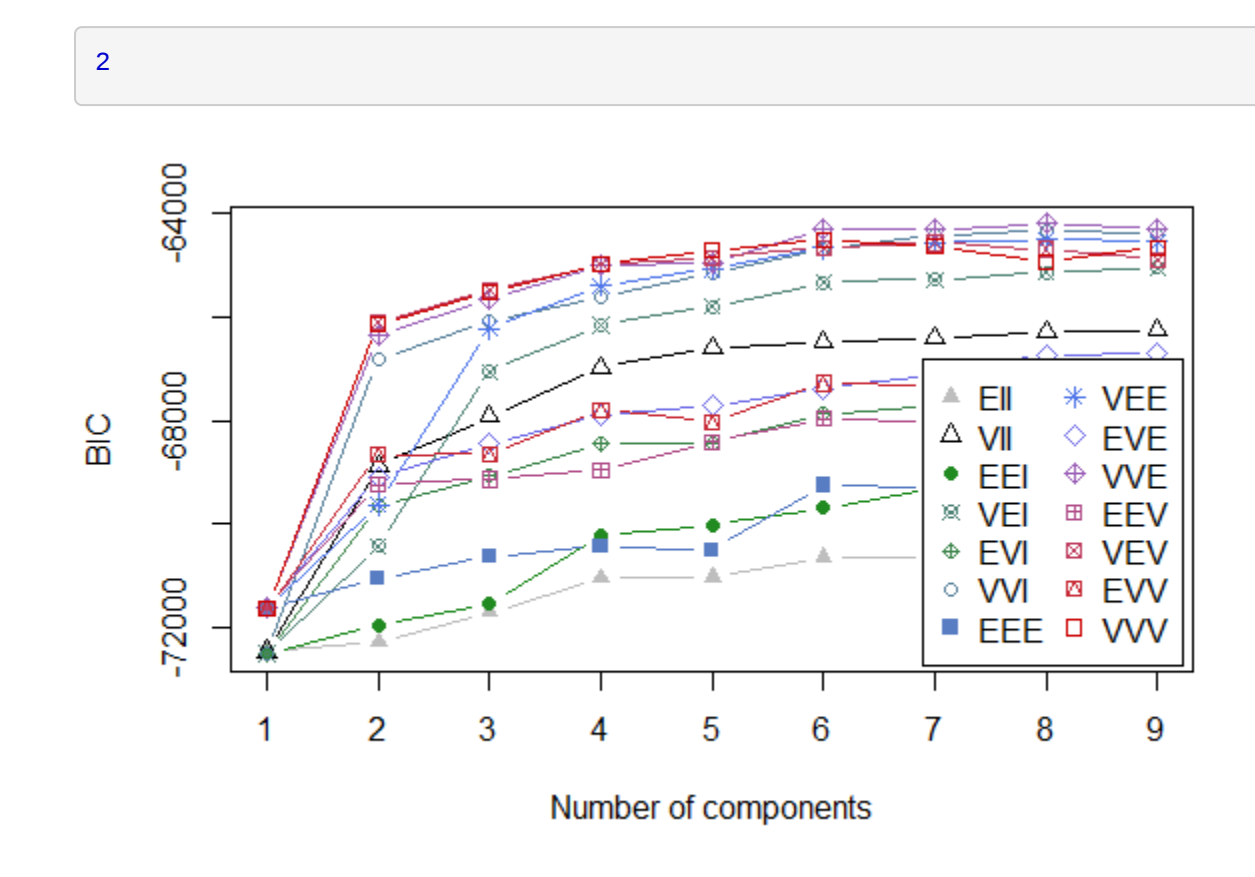
```
1
```

```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density
```

Hide

```
2
```

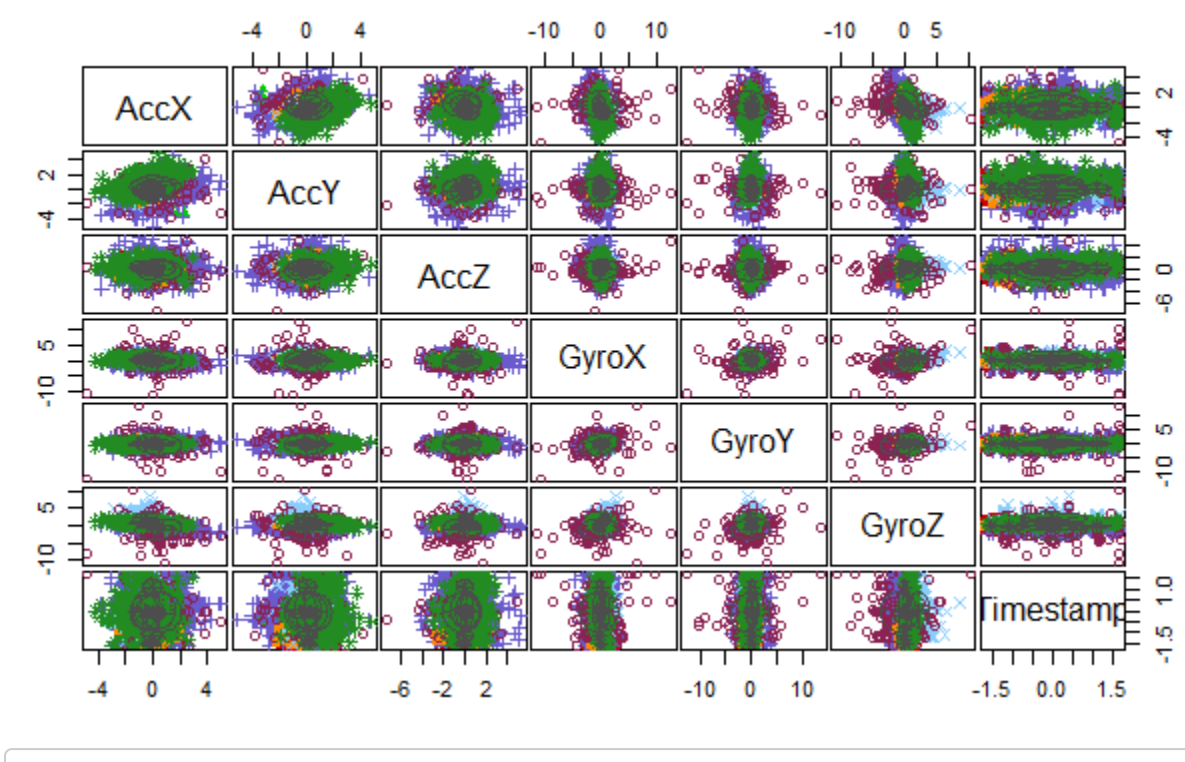



```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density
```

Hide

```
3
```

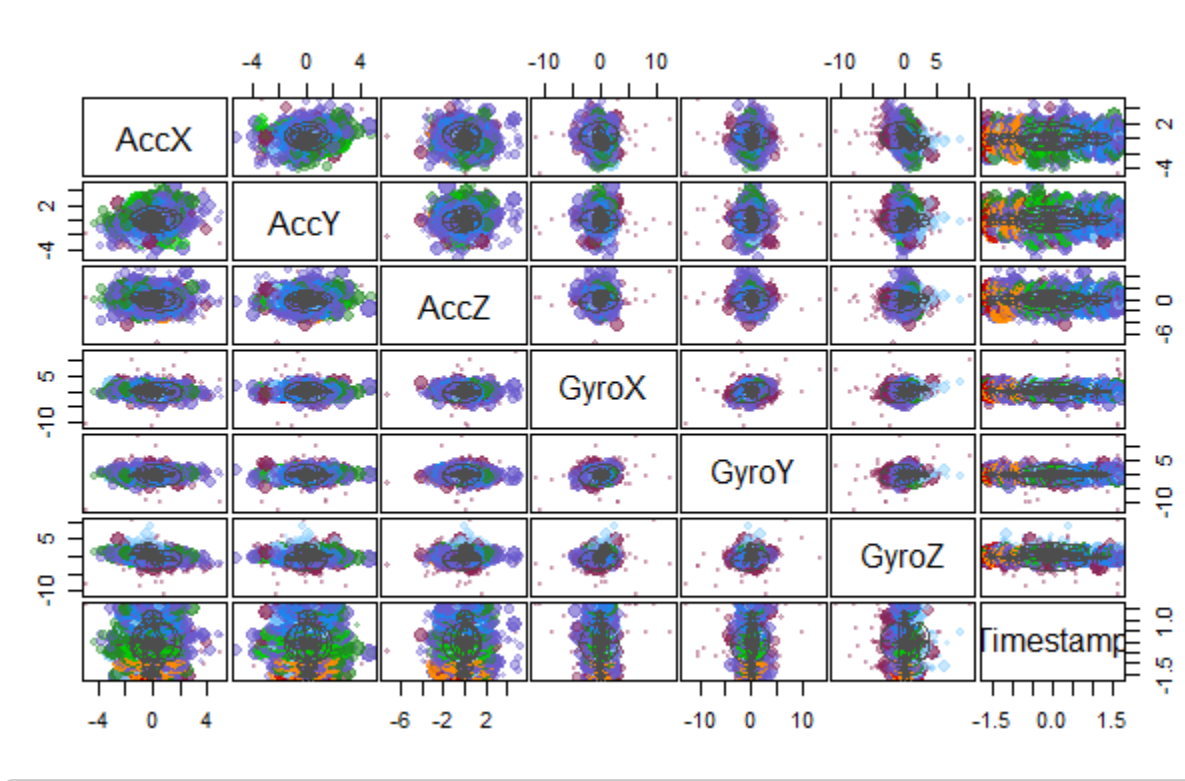



```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density
```

Hide

```
4
```

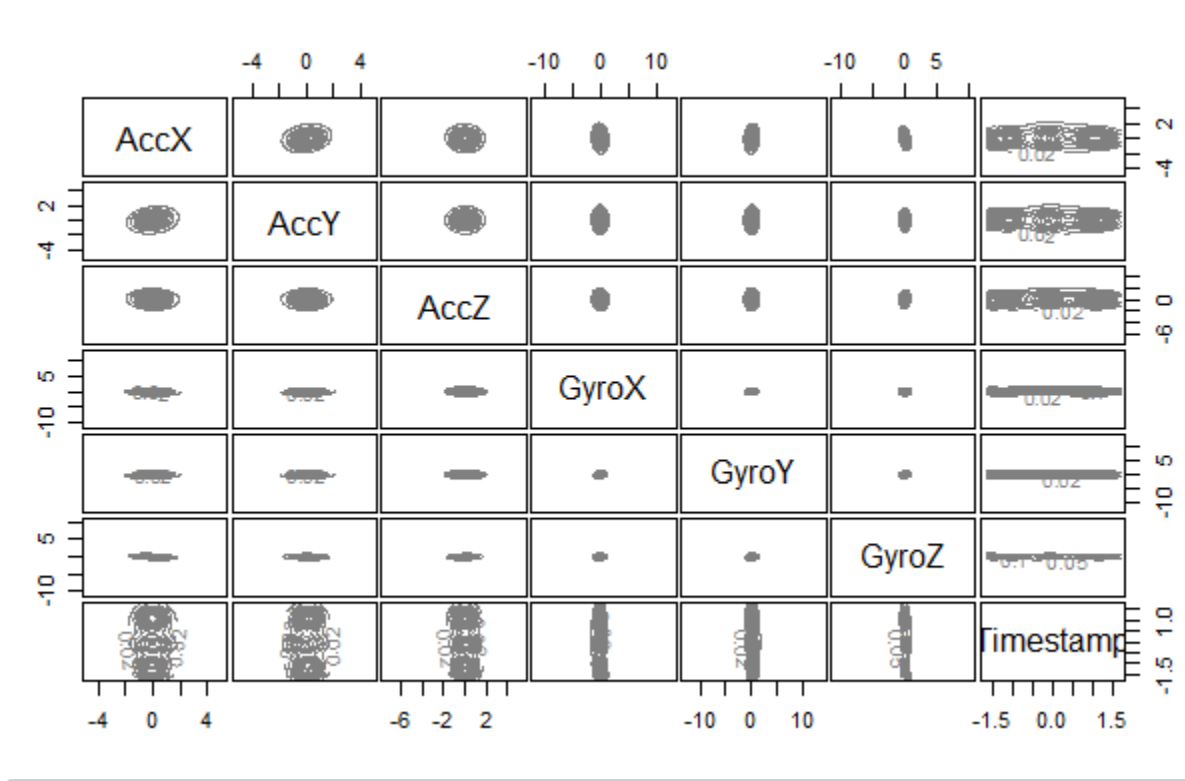



```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density
```

Hide

```
4
```

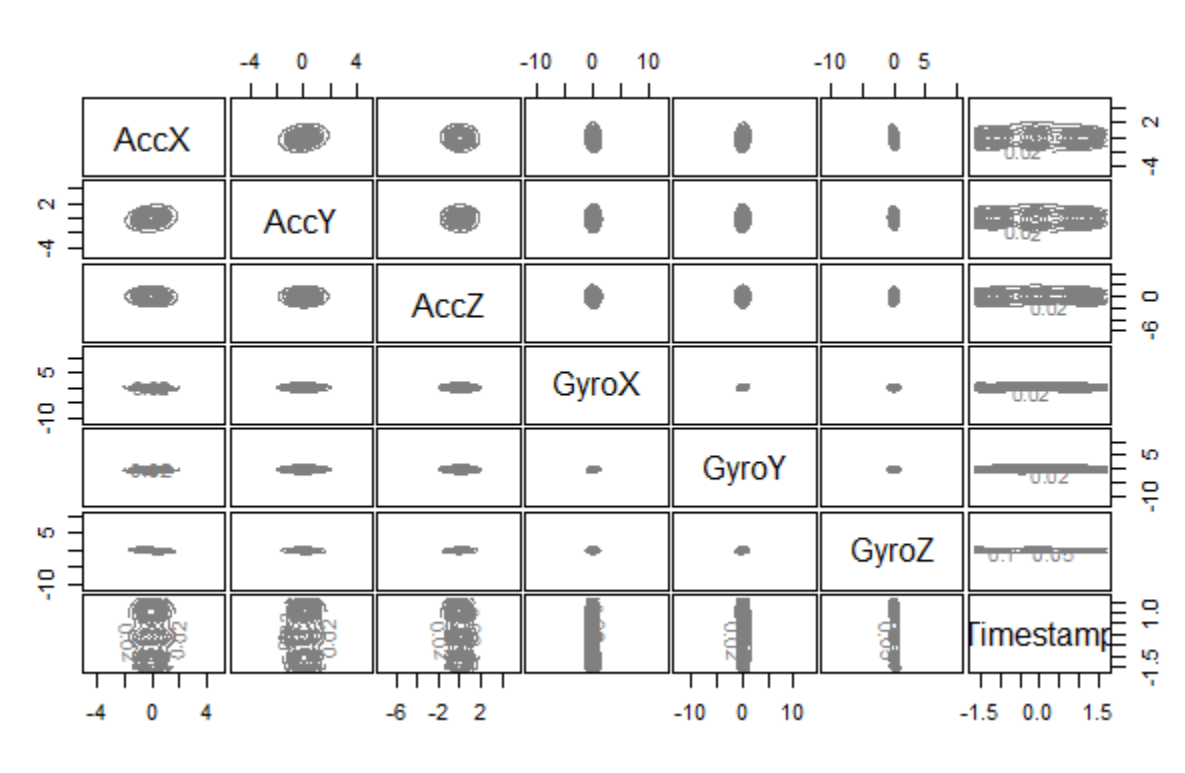



```
Model-based clustering plots:

1: BIC
2: classification
3: uncertainty
4: density
```

Hide

```
0
```



Comparison

Above are three approaches to clustering: kMeans, Hierarchical, and Model Based. Although powerful for exploring data that has an inherent hierarchy due to the visualization of all the branches of clusters, hierarchical clustering didn't fit the dataset well. This is due to the data used not having a hierarchical structure. kMeans did an acceptable job of separating the dataset into clusters, being able to cluster the data instances significantly better than random assignment. Most robust would likely be model based clustering, being able to cluster according to flexible structures. The algorithm was able to deduce an ellipsoidal, equal orientation clustering best fit the data.