Homework 2

Pattern Mining and Social Network Analysis

BOUYSSOU Gatien , de POURTALES Caroline, LAMBA Ankit

18 octobre, 2020

Contents

Supervised VS Unsupervised 2

Principal Components Analysis 3
- Proportion of variance explained (PVE) 3
- Deciding how many PCs to use 3
- Example 3
  - On R 3
  - On python with scikit-learn 7

Clustering 8
- K-means 8
  - Within-cluster variation (squared Euclidean distance) 8
  - K-means algorithm 8
  - Choice of k 8
  - Example 8
    - On R 8
    - On python with scikit-learn 8
- Hierarchical clustering 9
  - Interpreting a dendogram 9
  - Correlation-based distance 9
    - Euclidean distance 9
    - Correlation-based distance 9
  - Hierarchical clustering algorithm 9
  - Linkage 9
  - Example 10
    - On R 10
    - On python with scikit-learn 11

Validation techniques 12
- Bootstrapping 12
- Example 12
  - On R 12
  - On python with scikit-learn 12

Supervised VS Unsupervised

Principal Components Analysis

Proportion of variance explained (PVE)

Deciding how many PCs to use

Example

The following dataset consists of 40 tissue samples with measurements of 1,000 genes. The first 20 tissues come from healthy patients (H) and the remaining 20 come from a diseased patient group (D).

On R

```r
id <- "1VfVCQvWt121UN39NXZ4aR9Dmsbj-p9OU" # google file ID
GeneData <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",id), header = F)
colnames(GeneData)[1:20] = paste(rep("H", 20), c(1:20), sep = "")
colnames(GeneData)[21:40] = paste(rep("D", 20), c(1:20), sep = "")
row.names(GeneData) = paste(rep("G", 1000), c(1:1000), sep = "")
head(GeneData)
```

```
##             H1         H2         H3         H4         H5         H6
## G1 -0.96193340  0.4418028 -0.9750051  1.4175040  0.8188148  0.3162937
## G2 -0.29252570 -1.1392670  0.1958370 -1.2811210 -0.2514393  2.5119970
## G3  0.25878820 -0.9728448  0.5884858 -0.8002581 -1.8203980 -2.0589240
## G4 -1.15213200 -2.2131680 -0.8615249  0.6309253  0.9517719 -1.1657240
## G5  0.19578280  0.5933059  0.2829921  0.2471472  1.9786680 -0.8710180
## G6  0.03012394 -0.6910143 -0.4034258 -0.7298590 -0.3640986  1.1253490
##             H7          H8          H9        H10        H11        H12
## G1 -0.02496682 -0.06396600  0.03149702 -0.3503106 -0.7227299 -0.2819547
## G2 -0.92220620  0.05954277 -1.40964500 -0.6567122 -0.1157652  0.8259783
## G3 -0.06476437  1.59212400 -0.17311700 -0.1210874 -0.1875790 -1.5001630
## G4 -0.39155860  1.06361900 -0.35000900 -1.4890580 -0.2432189 -0.4330340
## G5 -0.98971500 -1.03225300 -1.10965400 -0.3851423  1.6509570 -1.7449090
## G6 -1.40404100 -0.80613040 -1.23792400  0.5776018 -0.2720642  2.1765620
##            H13         H14        H15        H16        H17        H18
## G1  1.33751500  0.70197980  1.0076160 -0.4653828  0.6385951  0.2867807
## G2  0.34644960 -0.56954860 -0.1315365  0.6902290 -0.9090382  1.3026420
## G3 -1.22873700  0.85598900  1.2498550 -0.8980815  0.8702058 -0.2252529
## G4 -0.03879128 -0.05789677 -1.3977620 -0.1561871 -2.7359820  0.7756169
## G5 -0.37888530 -0.67982610 -2.1315840 -0.2301718  0.4661243 -1.8004490
## G6  1.43640700 -1.02578100  0.2981582 -0.5559659  0.2046529 -1.1916480
##           H19         H20         D1         D2         D3         D4
## G1 -0.2270782 -0.22004520 -1.2425730 -0.1085056 -1.8642620 -0.5005122
## G2 -1.6726950 -0.52550400  0.7979700 -0.6897930  0.8995305  0.4285812
## G3  0.4502892  0.55144040  0.1462943  0.1297400  1.3042290 -1.6619080
## G4  0.6141562  2.01919400  1.0811390 -1.0766180 -0.2434181  0.5134822
## G5  0.6262904 -0.09772305 -0.2997108 -0.5295591 -2.0235670 -0.5108402
## G6  0.2350916  0.67096470  0.1307988  1.0689940  1.2309870  1.1344690
##             D5          D6         D7         D8          D9         D10
## G1 -1.32500800  1.06341100 -0.2963712 -0.1216457  0.08516605  0.62417640
## G2 -0.67611410 -0.53409490 -1.7325070 -1.6034470 -1.08362000  0.03342185
## G3 -1.63037600 -0.07742528  1.3061820  0.7926002  1.55946500 -0.68851160
## G4 -0.51285780  2.55167600 -2.3143010 -1.2764700 -1.22927100  1.43439600
## G5  0.04600274  1.26803000 -0.7439868  0.2231319  0.85846280  0.27472610
## G6  0.55636800 -0.35876640  1.0798650 -0.2064905 -0.00616453  0.16425470
##           D11          D12         D13        D14        D15        D16
## G1 -0.5095915 -0.216725500 -0.05550597 -0.4844491 -0.5215811  1.9491350
## G2  1.7007080  0.007289556  0.09906234  0.5638533 -0.2572752 -0.5817805
## G3 -0.6154720  0.009999363  0.94581000 -0.3185212 -0.1178895  0.6213662
## G4 -0.2842774  0.198945600 -0.09183320  0.3496279 -0.2989097  1.5136960
## G5 -0.6929984 -0.845707200 -0.17749680 -0.1664908  1.4831550 -1.6879460
## G6  1.1567370  0.241774500  0.08863952  0.1829540  0.9426771 -0.2096004
##            D17        D18         D19        D20
## G1  1.32433500  0.4681471  1.06110000  1.6559700
## G2 -0.16988710 -0.5423036  0.31293890 -1.2843770
## G3 -0.07076396  0.4016818 -0.01622713 -0.5265532
## G4  0.67118470  0.0108553 -1.04368900  1.6252750
## G5 -0.14142960  0.2007785 -0.67594210  2.2206110
## G6  0.53626210 -1.1852260 -0.42274760  0.6243603
```

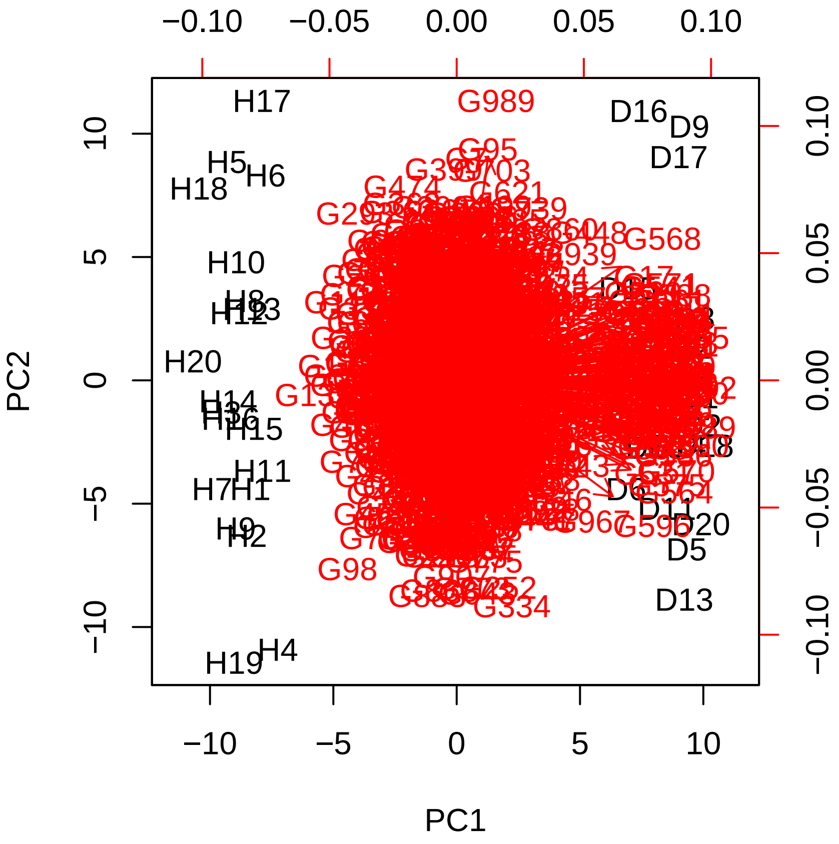
This matrix describes the “link” between each tissue sample and gene.

```
genematrix <- t(GeneData)
```

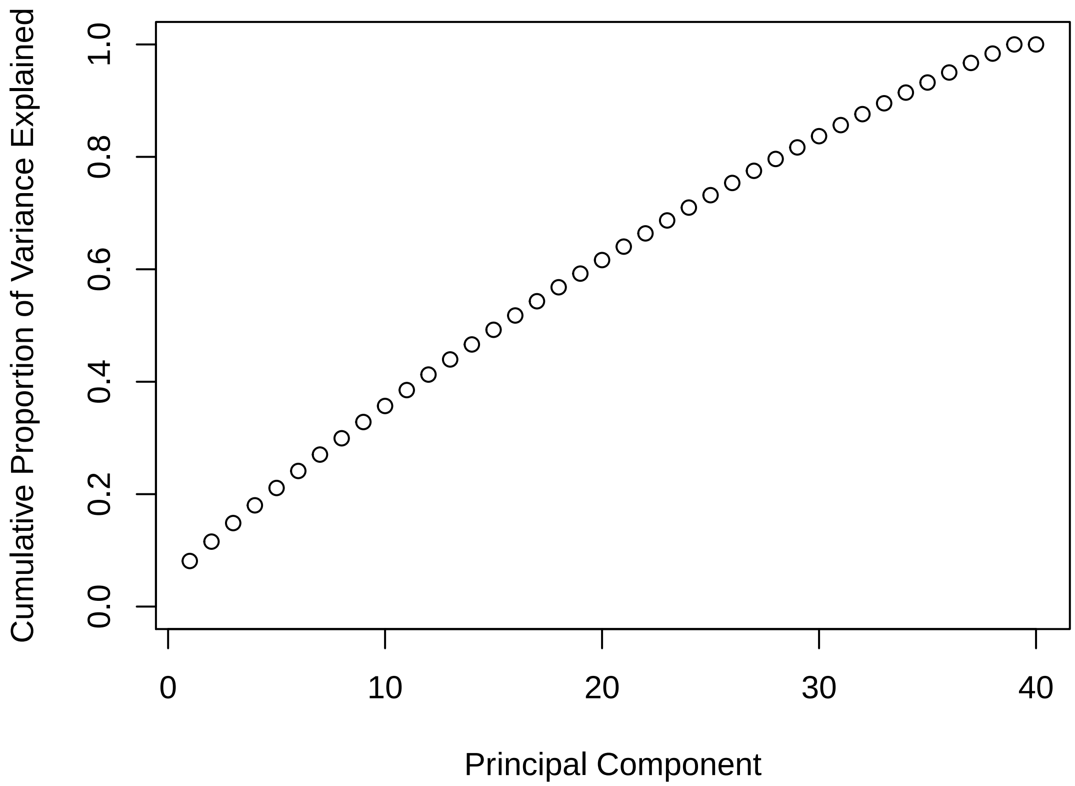
```
pca <- prcomp(genematrix, scale=TRUE)
summary(pca)
```

```
## Importance of components:
##                             PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     9.00460 5.87302 5.74347 5.61806 5.55344 5.50107 5.40069
## Proportion of Variance 0.08108 0.03449 0.03299 0.03156 0.03084 0.03026 0.02917
## Cumulative Proportion  0.08108 0.11558 0.14856 0.18013 0.21097 0.24123 0.27040
##                            PC8    PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     5.38575 5.3762 5.34146 5.31878 5.25016 5.18737 5.1667
## Proportion of Variance 0.02901 0.0289 0.02853 0.02829 0.02756 0.02691 0.0267
## Cumulative Proportion  0.29940 0.3283 0.35684 0.38513 0.41269 0.43960 0.4663
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     5.10384 5.04667 5.03288 4.98926 4.92635 4.90996 4.88803
## Proportion of Variance 0.02605 0.02547 0.02533 0.02489 0.02427 0.02411 0.02389
## Cumulative Proportion  0.49234 0.51781 0.54314 0.56803 0.59230 0.61641 0.64030
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     4.85159 4.79974 4.78202 4.70171 4.66105 4.64595 4.59194
## Proportion of Variance 0.02354 0.02304 0.02287 0.02211 0.02173 0.02158 0.02109
## Cumulative Proportion  0.66384 0.68688 0.70975 0.73185 0.75358 0.77516 0.79625
##                           PC29    PC30   PC31    PC32    PC33   PC34    PC35
## Standard deviation     4.53246 4.47381 4.4389 4.41670 4.39404 4.3591 4.23504
## Proportion of Variance 0.02054 0.02001 0.0197 0.01951 0.01931 0.0190 0.01794
## Cumulative Proportion  0.81679 0.83681 0.8565 0.87602 0.89533 0.9143 0.93226
##                          PC36    PC37   PC38    PC39     PC40
## Standard deviation     4.2184 4.12936 4.0738 4.03658 4.64e-15
## Proportion of Variance 0.0178 0.01705 0.0166 0.01629 0.00e+00
## Cumulative Proportion  0.9501 0.96711 0.9837 1.00000 1.00e+00
```

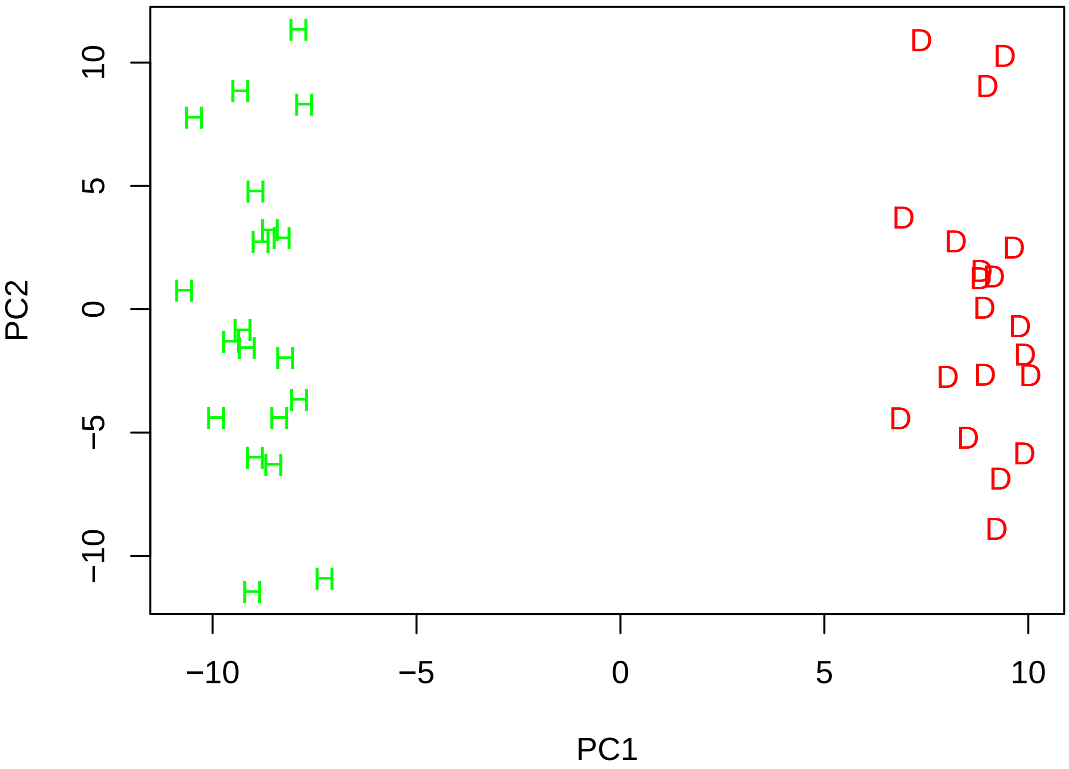
```
biplot(pca, scale=0)
```



```
pr.var=pca$sdev^2
pve=pr.var/sum(pr.var)
plot(cumsum(pve),
     xlab="Principal Component",
     ylab="Cumulative Proportion of Variance Explained",
     ylim=c(0,1),
     type='b')
```



```
p=plot(pca_plot$x[,1:2], type = "n")
p=p+points(pca_plot$x[0:20,1:2], pch = "H", col='green')
p=p+points(pca_plot$x[21:40,1:2], pch = "D",col='red')
```



On python with scikit-learn

Clustering

K-means

Within-cluster variation (squared Euclidean distance)

K-means algorithm

Choice of k

Example

```
k2 <- kmeans(genematrix, centers = 2, nstart = 15)
k2$cluster
```

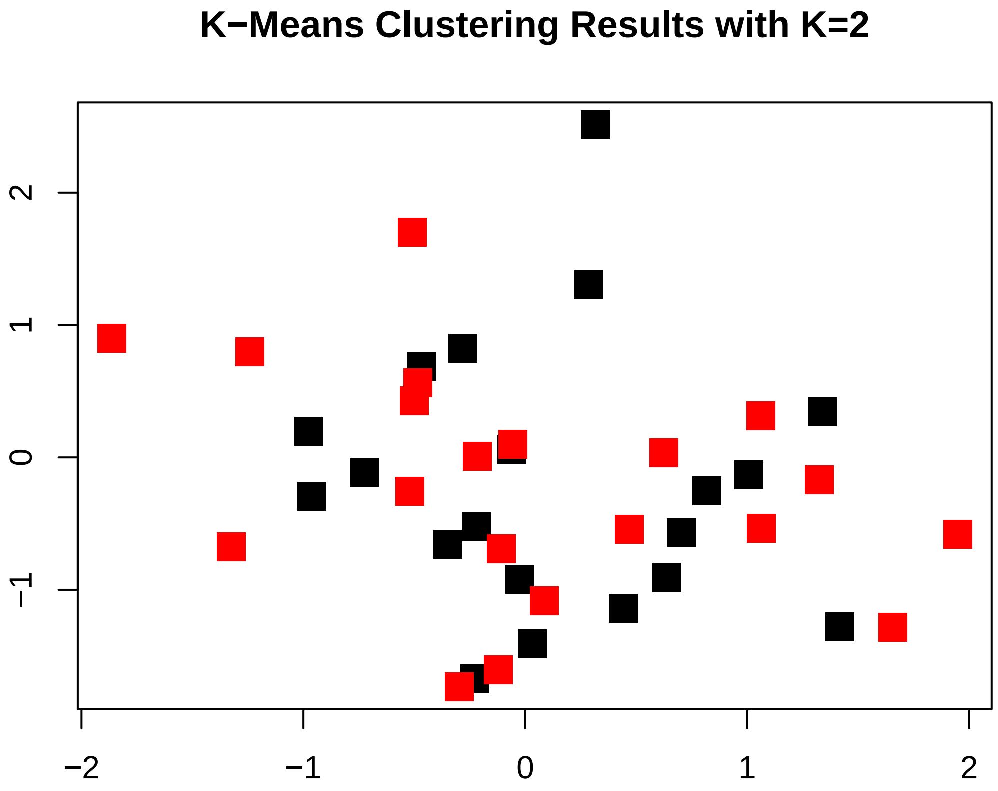
On R

```
##   H1  H2  H3  H4  H5  H6  H7  H8   H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##    1   1   1   1   1   1   1   1    1   1   1   1   1   1   1   1   1   1   1   1
##   D1  D2  D3  D4  D5  D6  D7  D8   D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##    2   2   2   2   2   2   2   2    2   2   2   2   2   2   2   2   2   2   2   2
```

```
k2$tot.withinss
```

```
## [1] 38366.84
```

```
p2= plot(genematrix,
     col=(k2$cluster),
     main="K-Means Clustering Results with K=2",
     xlab="", ylab="", pch=15, cex=2)
```



On python with scikit-learn

Hierarchical clustering

Interpreting a dendogram

Correlation-based distance

Euclidean distance

Correlation-based distance

Hierarchical clustering algorithm

Linkage

Example

```r
c_euclidean<-dist(genematrix, method = 'euclidean')
c_pearson <- cor(t(genematrix), method="pearson")
c_pearson <- as.dist(1-c_pearson)

#complete
clusters_complete_euclidean <- hclust(c_euclidean, method = "complete")
clusterCut <- cutree(clusters_complete_euclidean, 2)
clusterCut
```

On R

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```r
clusters_complete_correlation <- hclust(c_pearson, method = "complete")
clusterCut <- cutree(clusters_complete_correlation, 2)
clusterCut
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   2   1   1   2   1   2   1   2   2   1   1   2   1   1   2   2   2   2   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```r
#single
clusters_single_euclidean <- hclust(c_euclidean, method = "single")
clusterCut <- cutree(clusters_single_euclidean, 2)
clusterCut
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```r
clusters_single_correlation <- hclust(c_pearson, method = "single")
clusterCut <- cutree(clusters_single_correlation, 2)
clusterCut
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

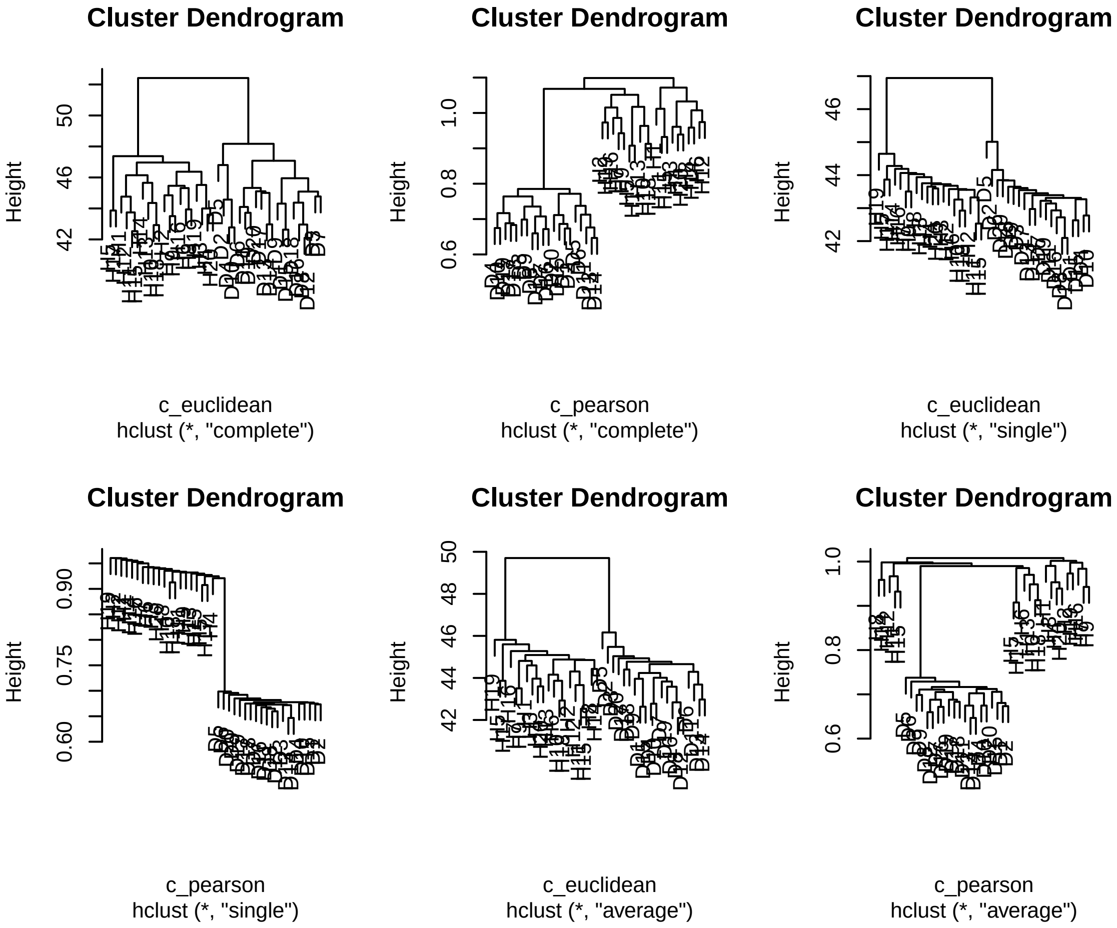
```r
#average
clusters_average_euclidean <- hclust(c_euclidean, method = "average")
clusterCut <- cutree(clusters_average_euclidean, 2)
clusterCut
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```r
clusters_average_correlation <- hclust(c_pearson, method = "average")
clusterCut <- cutree(clusters_average_correlation, 2)
clusterCut
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H13 H14 H15 H16 H17 H18 H19 H20
##   1   1   1   1   2   2   1   2   1   2   2   2   2   2   2   1   2   2   1   1
##  D1  D2  D3  D4  D5  D6  D7  D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18 D19 D20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```
par(mfrow = c(2, 3))
plot(clusters_complete_euclidean)
plot(clusters_complete_correlation)
plot(clusters_single_euclidean)
plot(clusters_single_correlation)
plot(clusters_average_euclidean)
plot(clusters_average_correlation)
```



We can see that the use of euclidean distance in the three methods (complete, single, average) gives good results (no missclassification) but the use of correlation-distance gives very bad results.

Furthermore all methods, except Average with correlation-distance, divide the graph in two groups (healthy and non-healthy) which is very good.

On python with scikit-learn

Validation techniques

Bootstrapping

Example

On R

On python with scikit-learn