Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering
IDP Institut für Datenanalyse
und Prozessdesign

HS 2016

Statistisches Data Mining (StDM)

Woche 3

Aufgabe 1 Non-Metrical MDS

a) Load the data set `voting.rda`. This data set has been taken from the HSAUR2 package and contains the number of times two congressmen voted differently on 19 enviromental bills in New Jersey. Use `isoMDS` and plot the results. The party of congressman can be obtained by

```
party = as.factor(c('R','R','D','D','R','R','R','D','D','D','D','R','R','D','D'))
```

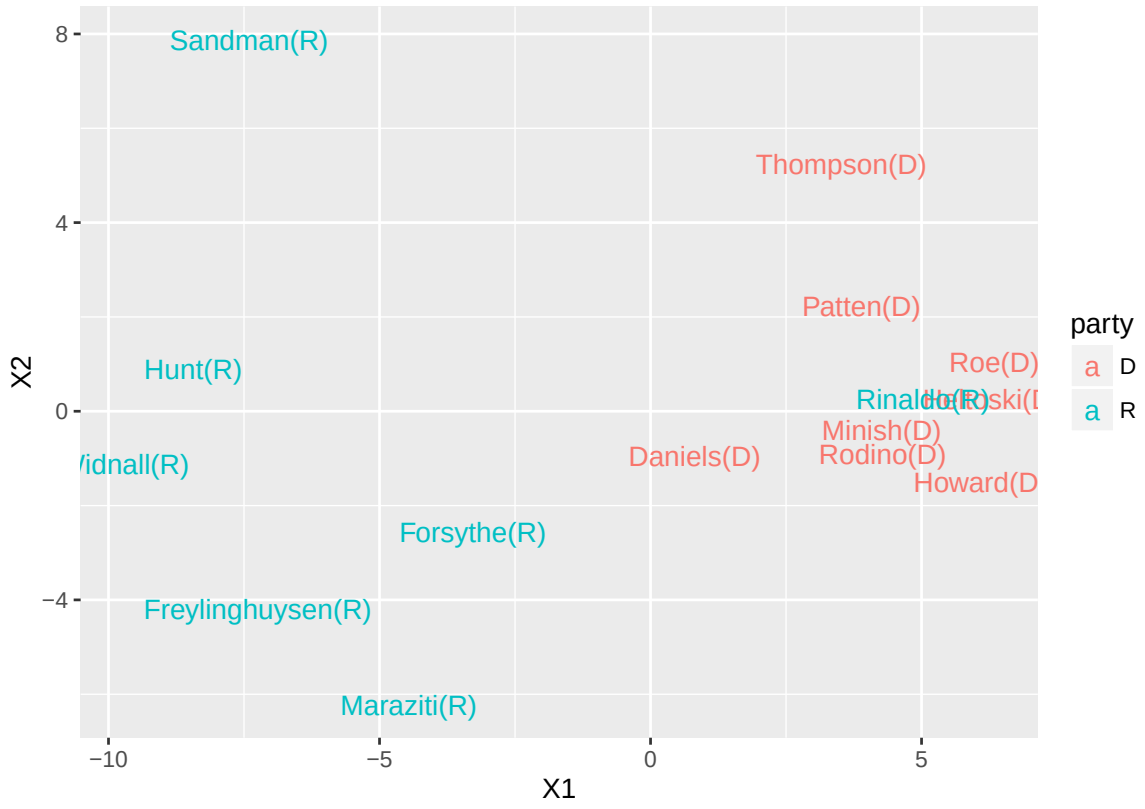
```
library(MASS)
library(ggplot2)
load(file.path(baseDir, 'voting.rda'))
party = as.factor(c('R','R','D','D','R','R','R','D','D','D','D','R','R','D','D'))
set.seed(1)
res = isoMDS(voting)
```

```
## initial  value 15.268246
## iter   5 value 10.264075
## final  value 9.879047
## converged
```

```
print(res$stress)
```

```
## [1] 9.879047
```

```
df = data.frame(res$points)
df$party = party
ggplot(df) + geom_text(aes(x=X1, y=X2,label=rownames(df), col=party))
```



b) To illustrate that only the order of the distances matter add 1 to each distance and then take the logarithm. Does the isoMDS significantly change?

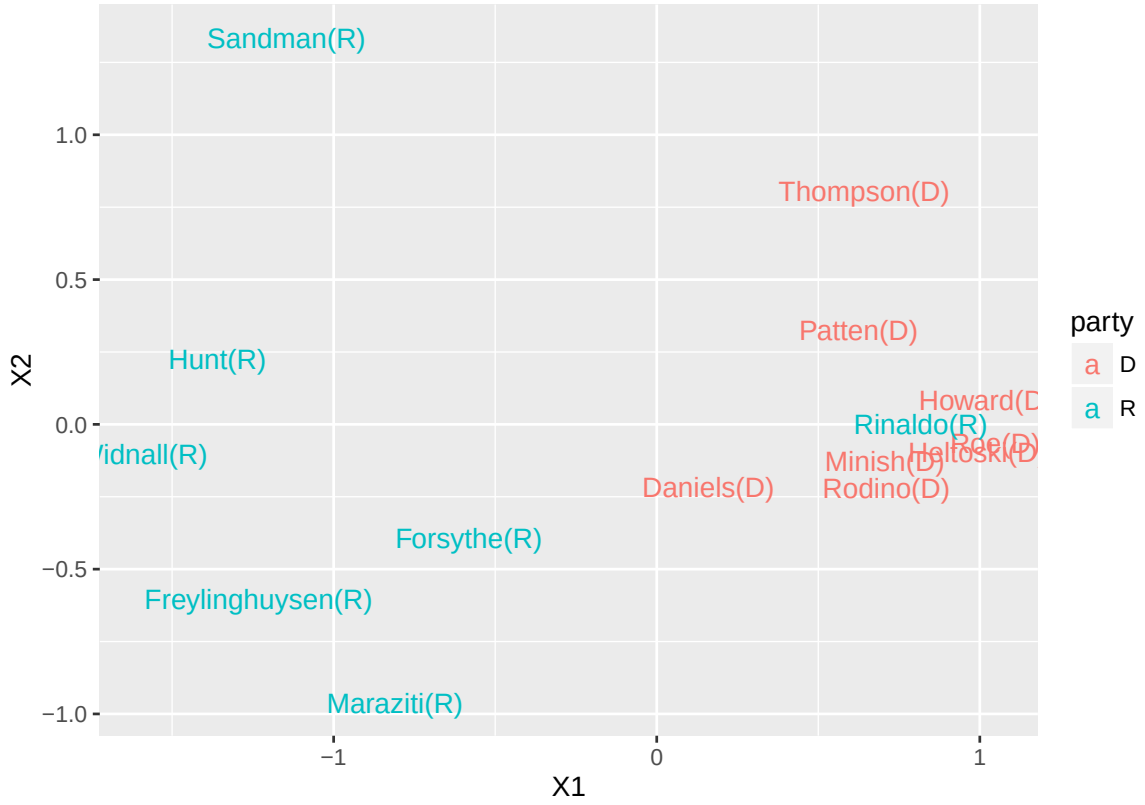
```
res = isoMDS(log(voting + 1))
```

```
## initial  value 17.971784
## iter   5 value 10.394217
## iter  10 value 10.244457
## iter  10 value 10.238156
## final  value 10.214085
## converged
```

```
print(res$stress)
```

```
## [1] 10.21408
```

```
df = data.frame(res$points)
df$party = party
ggplot(df) + geom_text(aes(x=X1, y=X2,label=rownames(df), col=party))
```



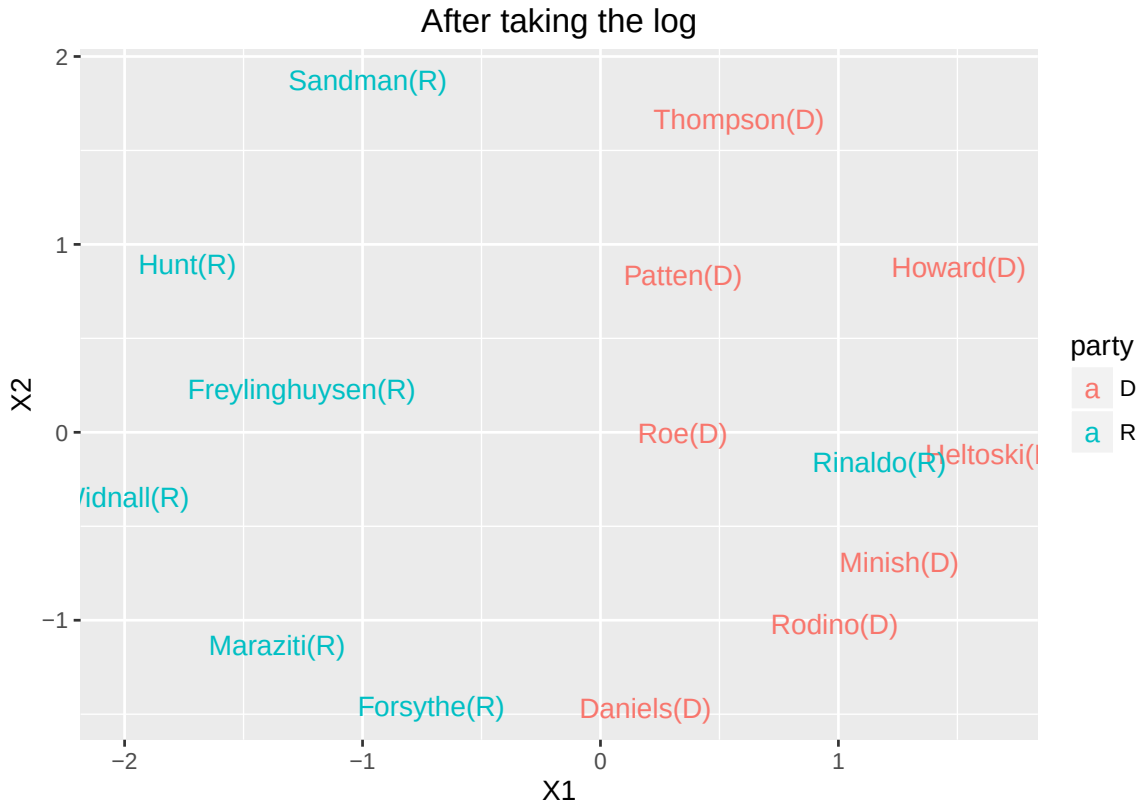
Since isoMDS is an interative numerical procedure the results are similar but not identical.

c) Repeat a) b) with Sammon Mapping

```
res = sammon(log(voting + 1))
```

```
## Initial stress        : 0.21542
## stress after  10 iters: 0.07669, magic = 0.342
## stress after  20 iters: 0.07298, magic = 0.500
## stress after  30 iters: 0.07290, magic = 0.500
```

```
df = data.frame(res$points)
df$party = party
ggplot(df) + geom_text(aes(x=X1, y=X2,label=rownames(df), col=party)) + ggtitle('After taking the
```



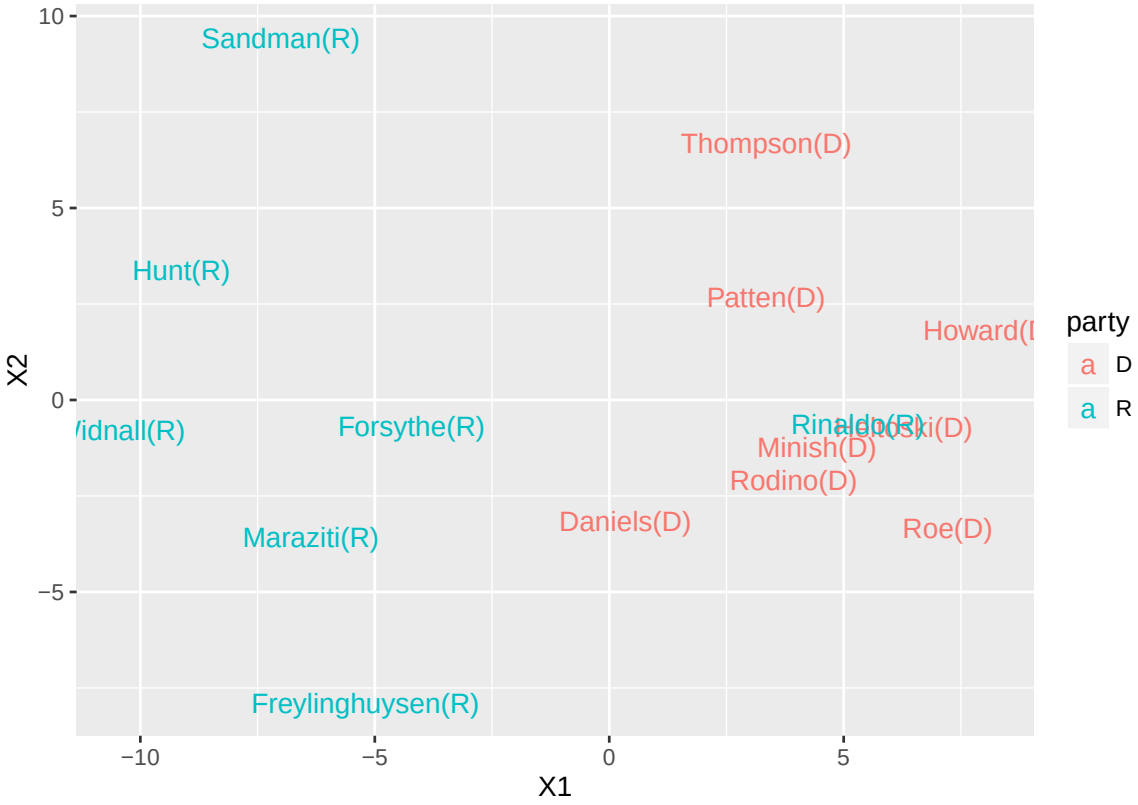

```
  res = sammon(voting)
```

```
## Initial stress        : 0.08813
## stress after  10 iters: 0.03087, magic = 0.500
## stress after  20 iters: 0.02889, magic = 0.500
## stress after  30 iters: 0.02843, magic = 0.500
## stress after  40 iters: 0.02835, magic = 0.500
```

```
  print(res$stress)
```

```
## [1] 0.02835305
```

```
  df = data.frame(res$points)
  df$party = party
  ggplot(df) + geom_text(aes(x=X1, y=X2,label=rownames(df), col=party))
```



d) Calculate the stress in the case of sammon mapping and compare against the result from c)

```
dS = dist(res$points)
sum( (dS  - voting)^2/voting) / sum(voting)
```

```
## [1] 0.02835305
```

e) To show the iterative nature of the `isoMDS` procedure, we start with a random initial configuration `Y = matrix(rnorm(2 * 15), ncol = 2)` and then perform only three iterations `isoMDS(voting, y = Y, maxit = 3)`. We plot the result and use it as a new starting point. We repeat this 10 times and look at the resulting plots. If you like, you can create an animation by storing the pngs and stick them together.

```
Y = matrix(rnorm(2 * 15), ncol = 2)
for (i in 1:30) {
  res = isoMDS(voting, y = Y, maxit = 1)
  print(res$stress)
  Y = res$points
  df = data.frame(Y)
  df$party = party
  plt = ggplot(df) + geom_text(aes(x=X1, y=X2,label=rownames(df), col=party)) + ggtitle(paste0(i,
  print(plt)
  ggsave(plt, file=paste0('tmp/g_',i,'.png'))
}

# Alternative Solution
```

```
if (FALSE) {
  library(ggvis)
  Y = matrix(rnorm(2 * 15), ncol = 2)
  i = 0
  data <- reactive({
    print(i)
    i <<- i + 1
    invalidateLater(100, NULL)
    res = isoMDS(voting, y = Y, maxit = 1)
    Y <<- res$points #I don't like this but this time there is no way around
    data = data.frame(
      x = Y[,1],
      y = Y[,2],
      party = party,
      stress = res$stress
    )
    data
  })
  data %>% ggvis(x =~ x, y =~ y, fill = ~party) %>%
    layer_points()
}
```

Note: Normaly `isoMDS` uses `cmdscale` as a starting point and therefore usually convergeces much faster.

Aufgabe 2 Visualizing Images

The file `training_48x48_aligned.gz` contains images and labels of the faces of several people. Use the following code to load the images, replace filename appropriately. If you like you can of course can create your pictures.

```
filename = file.path(baseDir,'training_48x48_aligned.gz')
dumm <- as.matrix(read.table(filename, sep=",", stringsAsFactors = FALSE))
X = dumm[,-1] #226 examples, 48^2 pixles
y = as.factor(dumm[,1]) #The label of the person from 0 to 5
N = sqrt(dim(X)[2])
par(mfrow=c(1,4))
par(mai=c(0.1,0.1,0.1,0.1))
for (i in c(1,50,100,200)) {
  m <- matrix(rev(X[i,]), nrow = N, ncol = N)
  image(m, useRaster = TRUE, axes = FALSE, col=gray((0:255)/255))
}
```

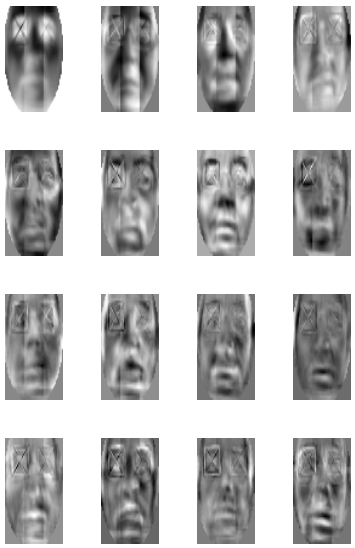
```
par(mfrow=c(1,1))
```

a) Sidetrack Eigenfaces (optional). Perform a PCA on the transposed matrix `X` and plot the first 16 `scores` as images. That is take the first, second, ... 2304 dimensional score vector and create an image `48x48` image using e.g. the `matrix` command as above.

```
require(ggplot2)
Xd = princomp(t(X))$scores
dim(Xd)
```

```
## [1] 2304  226
```

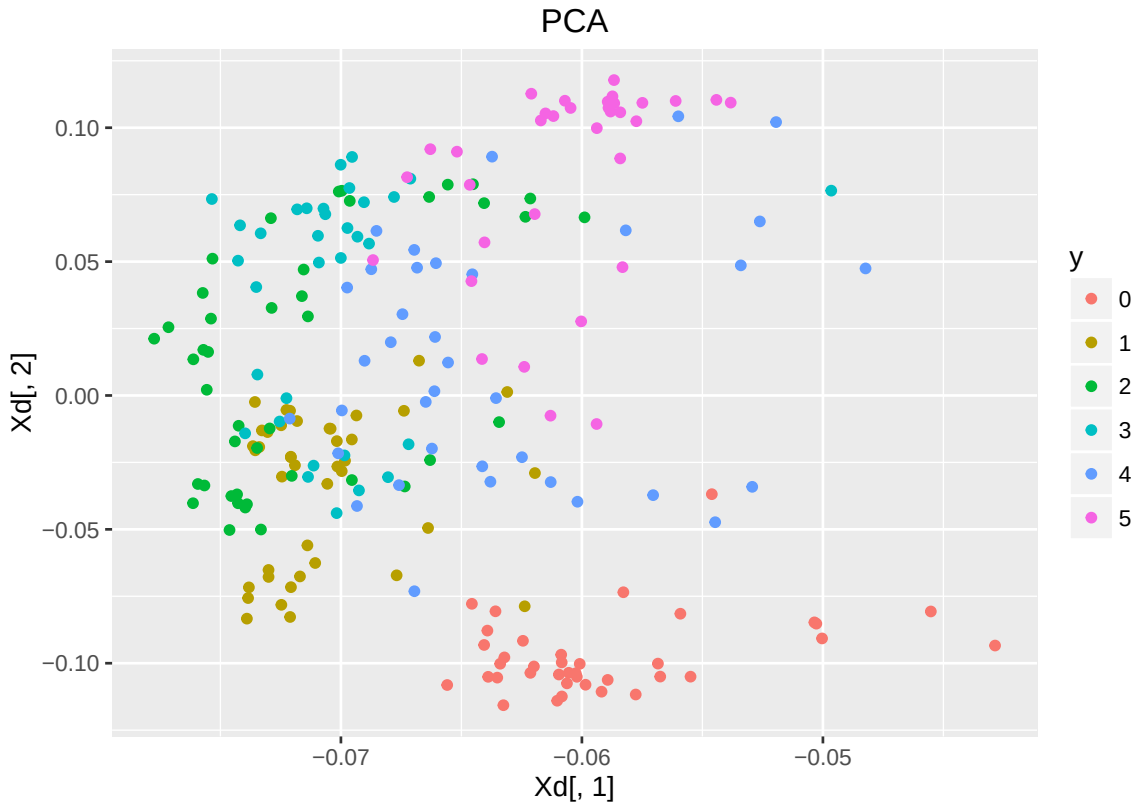
```
par(mfrow=c(4,4))
par(mai=c(0.1,0.1,0.1,0.1))
for (i in 1:16) {
  img = matrix(rev(Xd[,i]), nrow = 48, ncol = 48)
  image(img, useRaster = TRUE, axes = FALSE, col=gray((0:255)/255))
}
```



```
par(mfrow=c(1,1))
```

b) Now use the first 2 `loadings` of the PCA and plot them in a scatter plot together with we color by the lables `y`.

```
require(ggplot2)
 par(mfrow=c(1,1))
Xd = princomp(t(X))$loading
qplot(Xd[,1], Xd[,2], col=y) +  ggtitle('PCA')
```

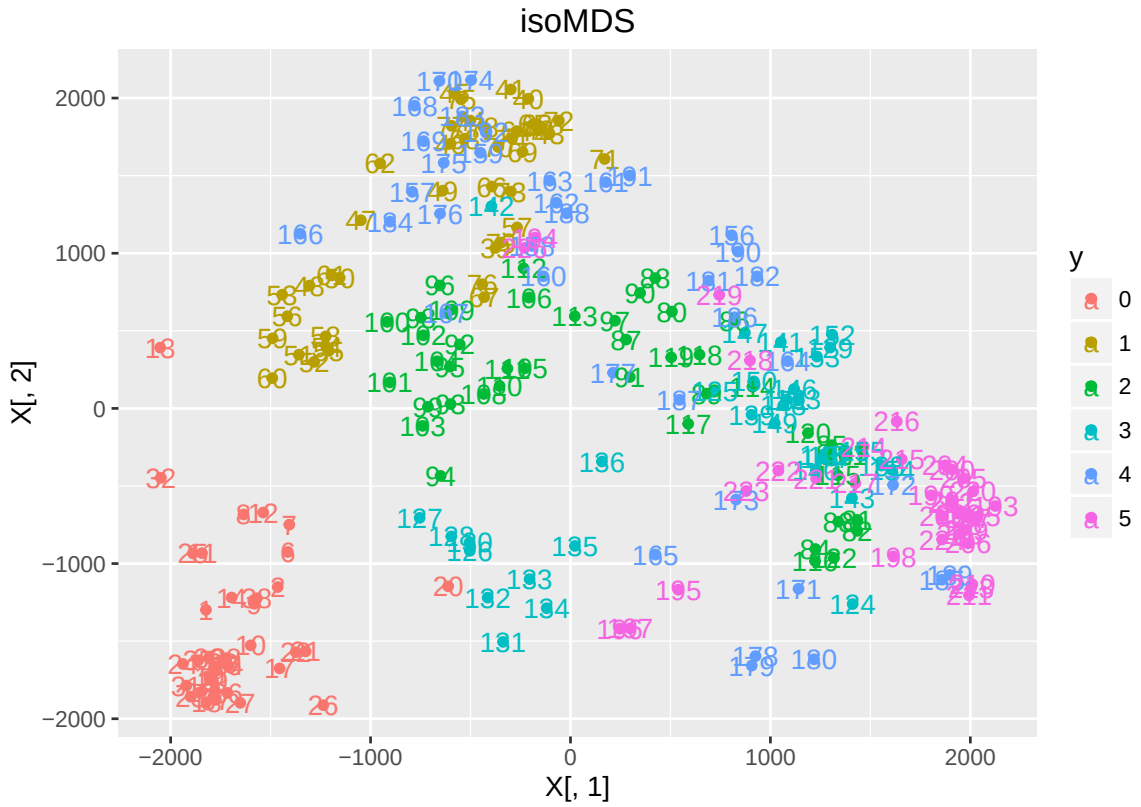


c) Now perform a non-Metrical MDS using the `isoMDS` and the `sannon` scaling, using euclidian distances, between the image.

```
library(MASS)
d = dist(X)
X <- isoMDS(d)$points
```

```
## initial  value 23.787009
## final  value 23.787005
## converged
```

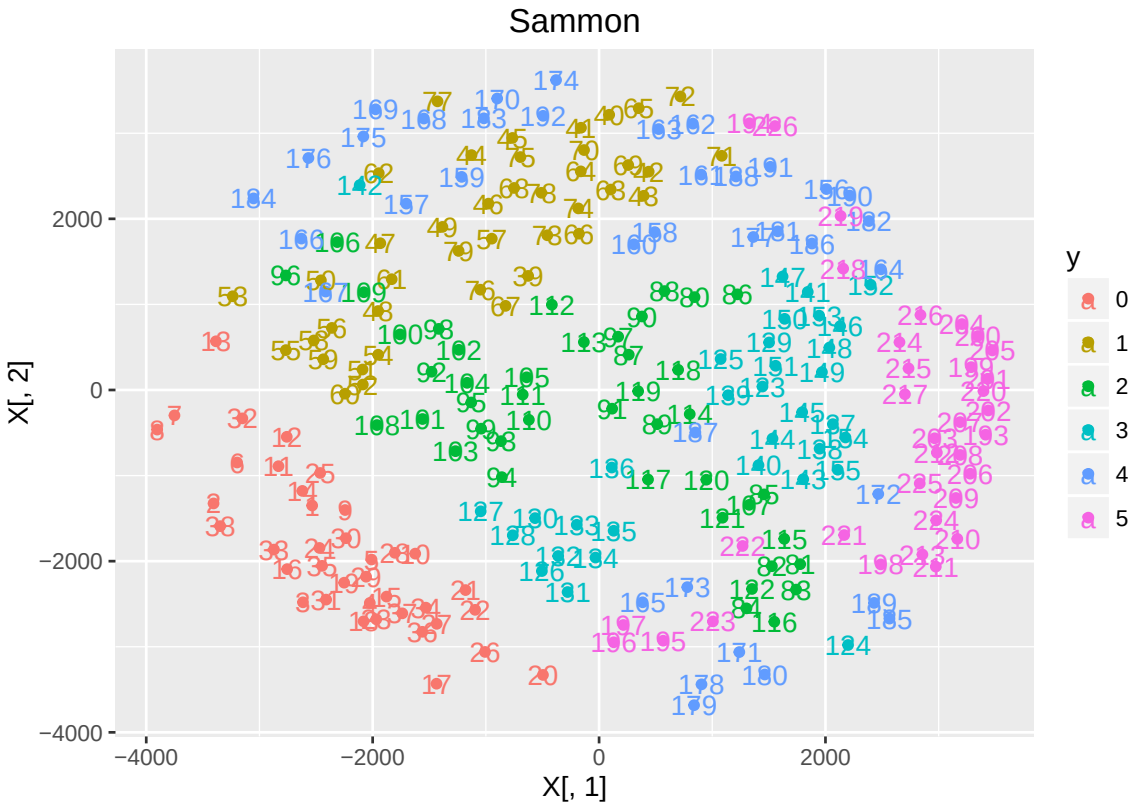
```
qplot(X[,1], X[,2], label=1:length(y),col=y)  +
  geom_text(alpha=1.0) + ggtitle('isoMDS')
```




```
  X <- sammon(d)$points
```

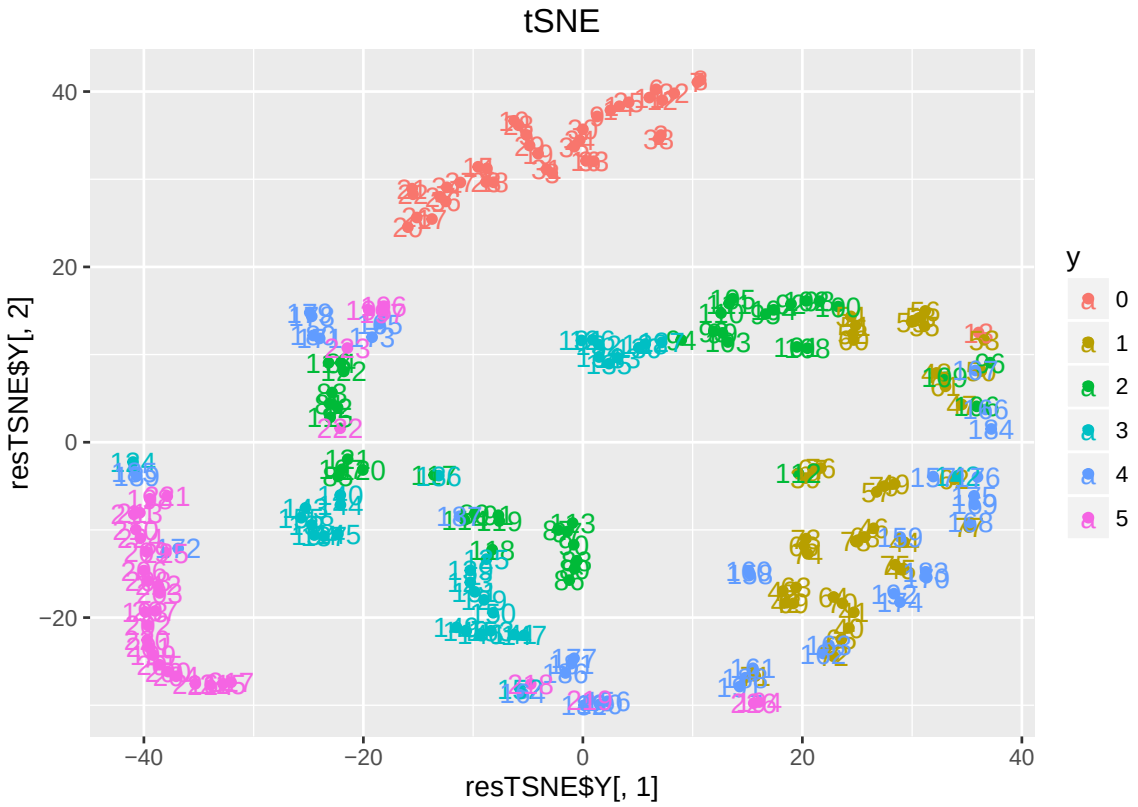
```
## Initial stress        : 0.26070
## stress after  10 iters: 0.19713, magic = 0.004
## stress after  20 iters: 0.15358, magic = 0.043
## stress after  30 iters: 0.10274, magic = 0.030
## stress after  40 iters: 0.10201, magic = 0.014
## stress after  50 iters: 0.10160, magic = 0.150
## stress after  60 iters: 0.10136, magic = 0.152
## stress after  70 iters: 0.10123, magic = 0.500
## stress after  80 iters: 0.10121, magic = 0.500
```

```
  qplot(X[,1], X[,2], label=1:length(y),col=y)  +
    geom_text(alpha=1.0) + ggtitle('Sammon')
```



d) Now perform a `tSNE` analysis using euclidian distances, between the image.

```
library(Rtsne)
d = dist(X)
resTSNE <- Rtsne(d, perplexity = 5)
qplot(resTSNE$Y[,1], resTSNE$Y[,2], label=1:length(y),col=y)  +
  geom_text(alpha=1.0) + ggtitle('tSNE')
```




```
if (FALSE) {
  # Some examples
  quartz()
  par(mfrow=c(1,4))
  par(mai=c(0.1,0.1,0.1,0.1))
  for (i in c(146,73, 69)) {
    m <- matrix(rev(X[i,]), nrow = N, ncol = N)
    image(m, useRaster = TRUE, axes = FALSE, col=gray((0:255)/255),main=i)
  }
}
```