Arbeitsblatt 3

Aufgabe 1: MNIST

Wir betrachten noch einmal den MNIST-Datensatz (`mnist_2k.Rdata`) mit 2000 zufällig ausgewählten handgeschriebenen Nummern von 0 bis 9 (Vergleiche Aufgabe 3 Arbeitsblatt2). Nach dem Laden des Datensatzes steht das Dataframe `x` zur Verfügung. Jede Zeile des Dataframes enthält die linearisierten Pixel-Grauwerte eines Bildes `pixel0` bis `pixel783`. Zudem gibt es eine Spalte `label` mit den Labeln.

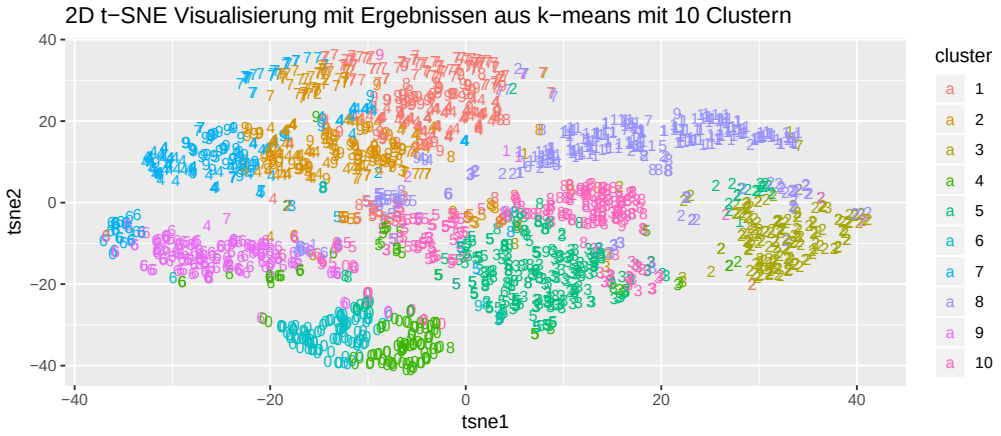
a) Führen Sie mit dem K-Means Ansatz mit 10 Clustern (`kmeans(...,centers=10)`) ein Clustering durch. Visualisieren Sie das Resultat im 2D t-SNE-Plot. Kommentieren Sie das Resultat. Code zur Visualisierung:

```
library(Rtsne)
restSNE <- Rtsne(x[, -1], perplexity = 20)
df <- as.data.frame(restSNE$Y)
names(df) <- c("tsne1","tsne2")
df$label <- as.factor(x$label)
df$cluster <- as.factor(res_cl$cluster) #res_cl -> Ergebnis kmeans
ggplot(data = df, aes(x = tsne1, y = tsne2, col = cluster, label=label)) +
  geom_text(size=3)
```

```
load('Daten/mnist_2k.Rdata')
set.seed(79)
res_cl <- kmeans(x[, -1], centers=10)
res_cl$tot.withinss
```

```
[1] 5128826304
```

```
library(Rtsne)
set.seed(1)
restSNE <- Rtsne(x[, -1], perplexity = 20)
df <- as.data.frame(restSNE$Y)
names(df) <- c("tsne1","tsne2")
df$label <- as.factor(x$label)
df$cluster <- as.factor(res_cl$cluster)
library(ggplot2)
ggplot(data = df, aes(x = tsne1, y = tsne2, col=cluster, label=label)) +
  geom_text(size=3) +
  ggtitle("2D t-SNE Visualisierung mit Ergebnissen aus k-means mit 10 Clustern")
```



Das Ergebnis des K-means Clustering mit k=10 sieht ziemlich gut aus. Die 1er werden z.B. ziemlich gut abgetrennt. Allgemein entsprechen die Cluster jedoch nicht exakt den 10 Nummern. Gewisse Nummern wie z.B. die 0er sind in zwei Clustern aufgeteilt. Auf der anderen Seite sind gewisse Kombinationen z.B. 4er, 7er und 9er oder 3er, 5er und 8er in jeweils zwei Clustern vermischt. Im Vergleich zur t-SNE Separierung ist das Ergebnis schlechter. Das genaue Ergebnis ist vom gewählten set.seed abhängig.

b) Erzeugen Sie mit dem Befehl `table(df$label, LETTERS[df$cluster])` eine Konfusionsmatrix. Was sehen wir? Was sagen uns die Zahlen?

```
table(df$label, LETTERS[df$cluster] )
```

	A	B	C	D	E	F	G	H	I	J
0	0	0	0	80	9	91	2	0	6	8
1	0	1	1	0	1	0	0	211	2	1
2	3	2	159	5	20	0	2	41	3	2
3	1	5	3	6	110	1	1	12	1	33
4	69	58	0	0	0	0	72	2	2	0
5	15	9	0	14	55	2	5	27	2	58
6	0	1	2	13	1	2	22	15	146	9
7	93	59	2	0	0	0	30	8	1	1
8	12	10	1	2	48	2	2	10	0	97
9	59	82	0	1	2	1	48	3	0	2

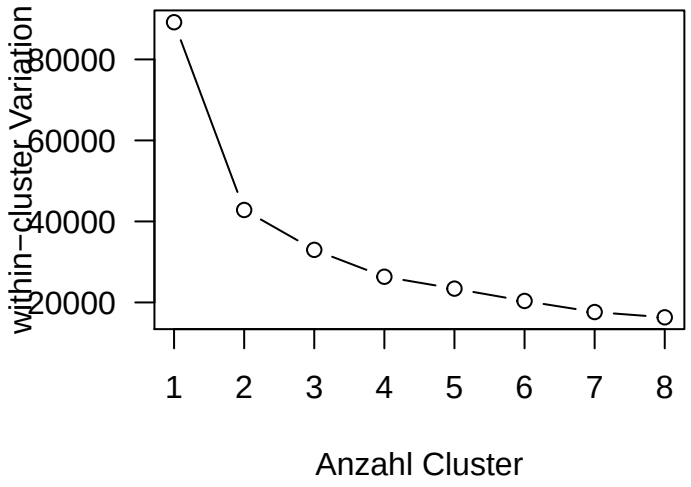
Die Konfusionsmatrix ist eine Kreuztabelle zwischen den zugewiesenen Clustern (Spalten) aus der Clusteranalyse und den effektiven Labels (Zeilen). Darin kann man die Qualität des Clustering übersichtlich ablesen. Grundsätzlich ist die Konfusionsmatrix ein Tool aus dem überwachten Lernen, da wir beim unüberwachten Lernen (Clustering) die Labels in der Praxis ja nicht kennen.
Man sieht zum Beispiel, dass die 0en hauptsächlich in den Clustern D und F zu finden sind. Oder, dass die 1er ziemlich gut durch das Cluster H abgebildet werden. Achtung: Auswertung vom set.seed abhängig.

Aufgabe 2: Abstimmungen

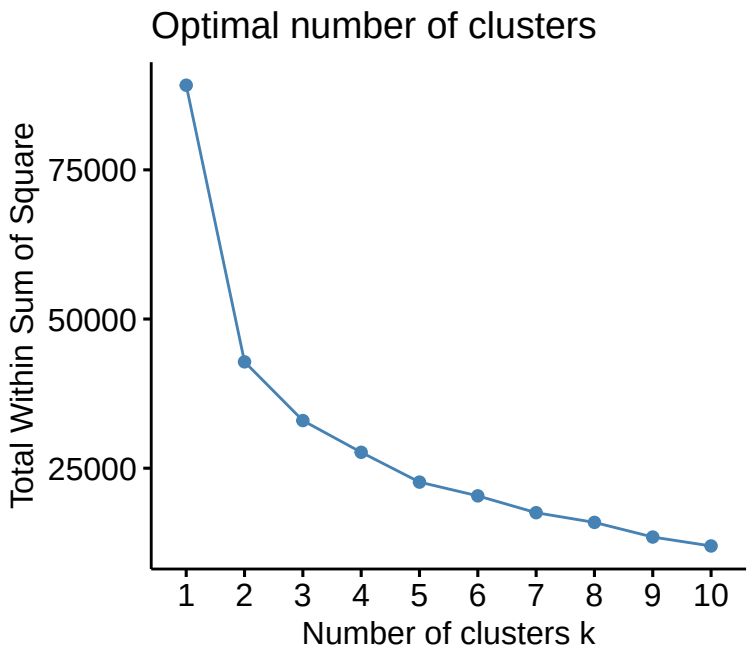
Wir wollen nun noch einmal mit dem Datensatz der eidgenössischen Abstimmungen (`abst.Rdata`) arbeiten.

a) Wir versuchen nun die Kantone zu clustern. Verwenden Sie dazu den K-means Algorithmus. Bestimmen Sie die optimale Anzahl Cluster mit Hilfe der “Within-cluster Variation”.

```
load("Daten/abst.Rdata")
set.seed(7)
## Bestimmen der optimalen Clusteranzahl
wcv <-  rep(0, 8) # Initialisierung
for (i in 1:8)
  wcv[i] <- sum(kmeans(abst, centers = i)$withinss)
plot(1:8, wcv, type = "b", xlab = "Anzahl Cluster",
     ylab = "within-cluster Variation", las=1)
```



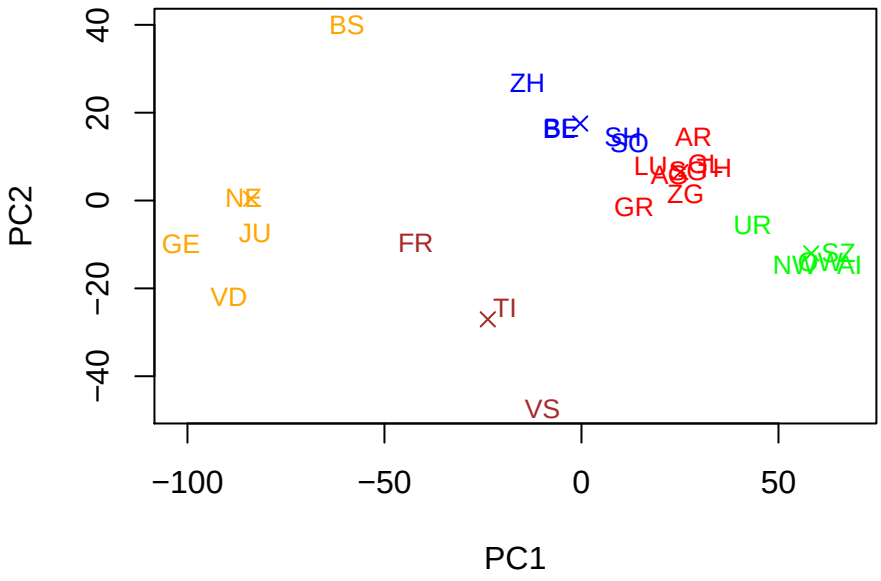

```
#Alternative
library(factoextra)
fviz_nbclust(abst, kmeans, method="wss")
```



Die korrekte Anzahl Cluster ist hier nicht eindeutig bestimmbar. Knicke sind bei 2, 5 und 7 Clustern zu finden. In der Lösung wird mit 5 Cluster weitere gearbeitet. 2 oder 7 Cluster wären aber allenfalls auch möglich.

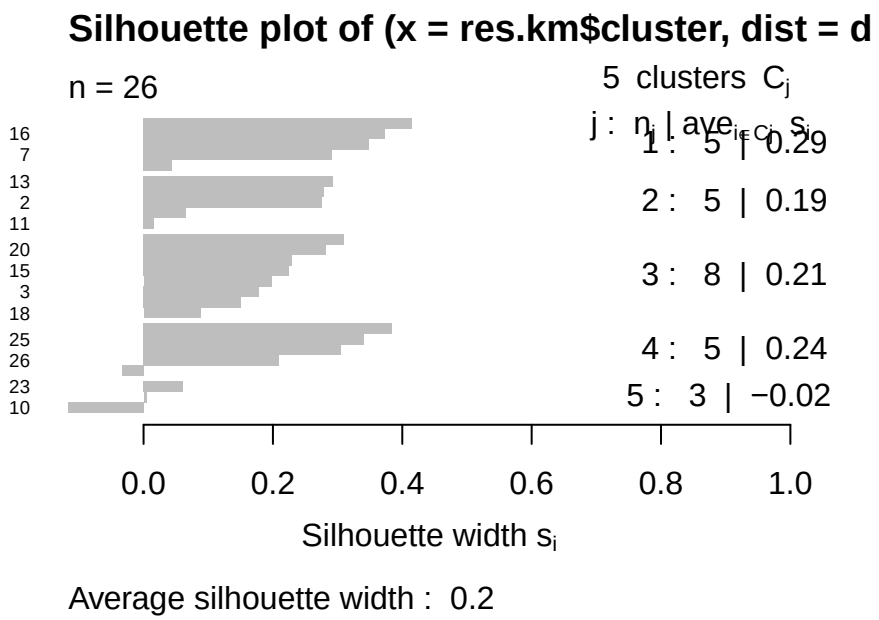
b) Visualisieren Sie das Ergebnis der Clusteranalyse in einem 2D-Plot der ersten beiden Hauptkomponenten einer PCA. Im Standardplot können Sie dazu die Cluster farblich wie folgt übergeben (`col=c("green", "blue", ...)[res.km$cluster]`). Häufig ist es informativ, wenn man zusätzlich die Zentren der Cluster plottet. Dazu muss man die Daten in die PCA-Projektion transformieren (`predict(abst.pca, newdata=res.km$centers)`).

```
res.km <- kmeans(abst, 5)
## Visualisierung mit PCA
abst.pca <- prcomp(abst, scale=FALSE)
plot(abst.pca$x[,1], abst.pca$x[,2], type="n", xlab="PC1", ylab="PC2")
text(abst.pca$x[,1], abst.pca$x[,2], labels=row.names(abst), cex=0.8,
     col=c("green", "blue", "red", "orange", "brown")[res.km$cluster])
points(predict(abst.pca, newdata=res.km$centers),
       col=c("green", "blue", "red", "orange", "brown"), pch=4)
```

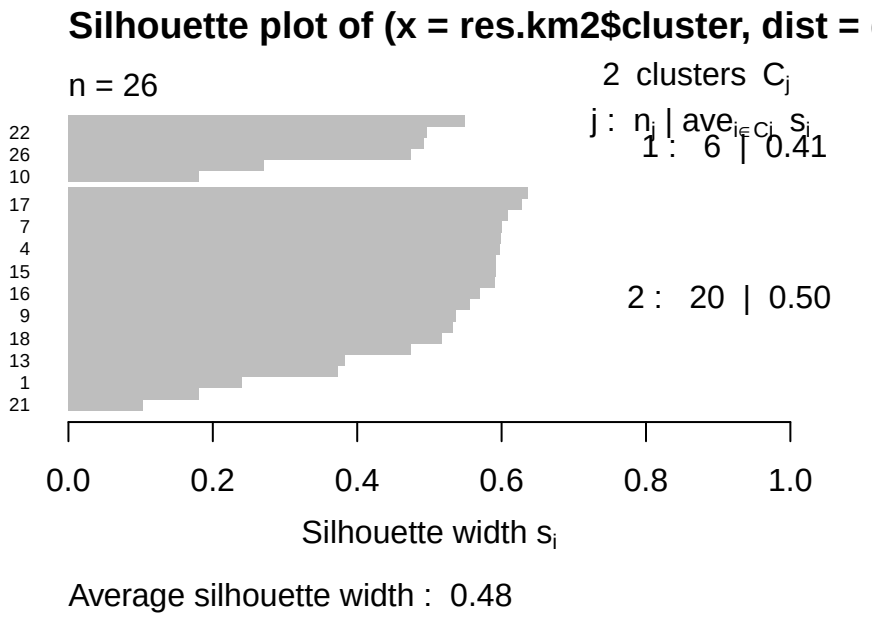


c) Analysieren Sie Ihr Ergebnis mit einem Silhouetten-Plot `plot(silhouette(...))`. Dazu müssen Sie vorgängig das Paket `cluster` laden (`library(cluster)`).

```
library(cluster)
plot(silhouette(x=res.km$cluster, dist=dist(abst)), cex.names=0.6)
```



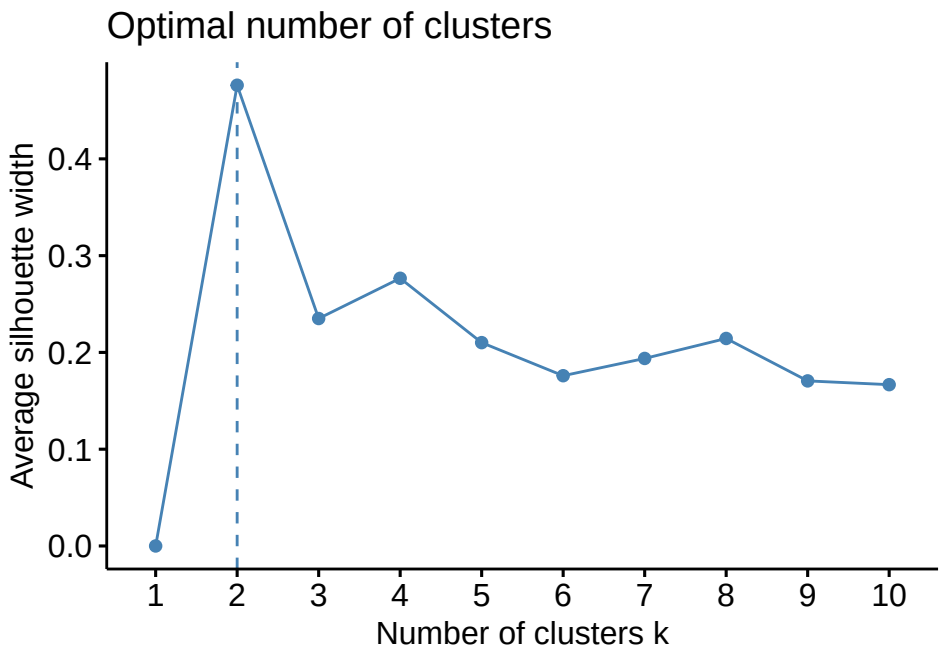
```
## mit nur 2 Clustern
res.km2 <- kmeans(abst, 2)
plot(silhouette(x=res.km2$cluster, dist=dist(abst)),  cex.names=0.6)
```



Bei 5 Clustern ist die Struktur schwach. Werden nur 2 Cluster verwendet, ist die Struktur besser.

d) Bestimmen Sie die optimale Anzahl Cluster mit mittlerer Silhouttenbreite. (Tipp: R-Funktion `fviz_nbclust(...,method='silhouette')` aus dem Paket `factoextra`).

```
library(factoextra)
fviz_nbclust(abst, kmeans, method="silhouette")
```



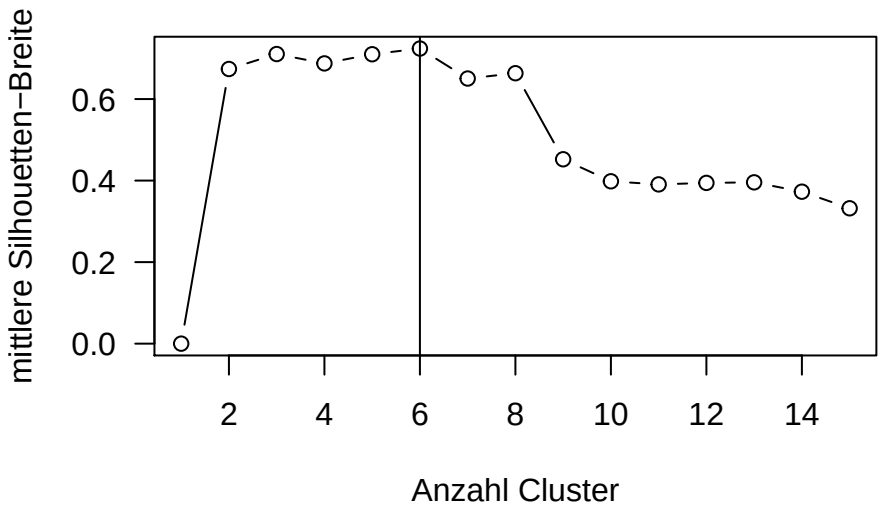
Die Silhouttenanalyse spricht für 2 Cluster.

Aufgabe 3: Nationalrat

In dieser Aufgabe arbeiten wir noch einmal mit dem Abstimmungsverhalten des Schweizer Parlaments (Nationalrats). Datensatz `voting_NR.rdata` mit der "Distanz"-Matrix: `NR_voting` und den Metainformationen (Fraktion und Kanton): `NR_meta` (vergleiche Arbeitsblatt 2 Aufgabe 2).

a) Führen Sie mit dem K-Medoids/PAM Ansatz das Clustering für die Distanzmatrix `NR_voting` durch. Wie viele Klassen sind hier angebracht?

```
load("Daten/voting_NR.Rdata")
set.seed(10)
sil <-  rep(0, 15) # Initializierung
for (i in 2:15) sil[i] <- summary(silhouette(x=pam(NR_voting, k=i)$cluster,
                                             dist=NR_voting))$avg.width
plot(x=1:15, sil, type = "b", xlab = "Anzahl Cluster",
     ylab = "mittlere Silhouetten-Breite", las=1)
abline(v=which.max(sil))
```




```
## rund 6 Cluster sind hier angebracht
res.pam <- pam(NR_voting, k=6)
#Alternative
library(fpc)
res.pamk <- pamk(NR_voting)
```

Rund 6 Cluster sind hier angebracht.

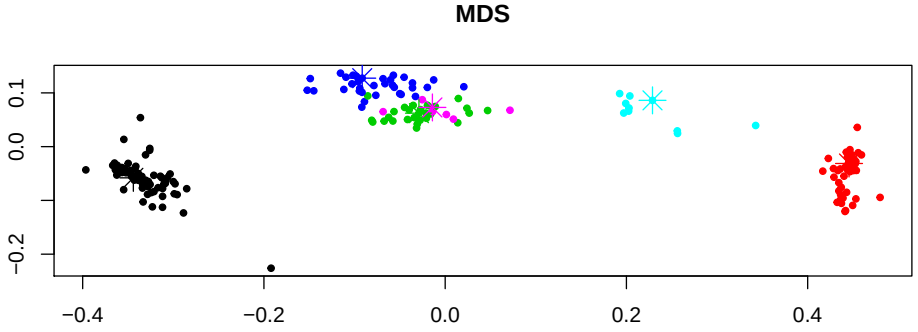
b) Was sind die repräsentativen Mitglieder der Cluster (Medoid)?

```
res.pam$medoids
```

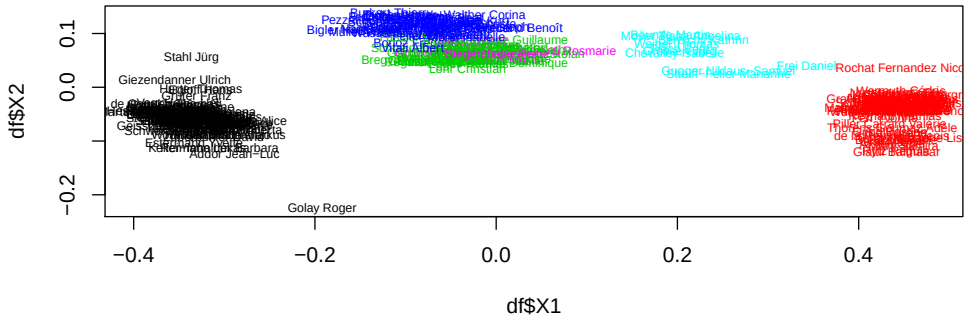
```
[1] "Tuena Mauro"      "Munz Martina"     "Müller Leo"       "Gössi Petra"
[5] "Bertschy Kathrin" "Campell Duri"
```

c) Visualisieren Sie das Ergebnis der Clusteranalyse mit MDS (`cmdscale`) oder t-SNE (`Rtsne`).

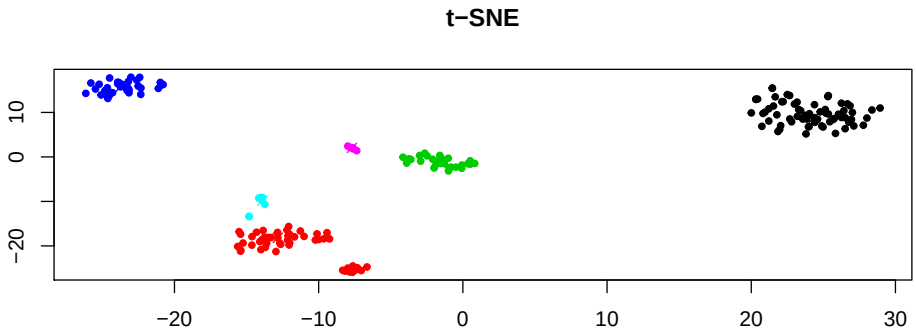
```
set.seed(20)
##MDS
res.cmd <- cmdscale(NR_voting)
plot(res.cmd, pch=20, main="MDS", xlab="", ylab="", col=res.pam$clustering)
points(res.cmd[res.pam$medoids,], pch=8, col=1:6, cex=2)
```




```
res <- cmdscale(NR_voting)
df <- data.frame(res)
plot(df$X1, df$X2, type = 'n')
text(df$X1, df$X2, label=rownames(df), cex=0.6, col=res.pam$clustering)
```




```
##t-SNE
tsne_dist <- Rtsne(NR_voting, PCA=FALSE, is_distance=TRUE, perplexity=15, max_iter=3000)
plot(tsne_dist$Y, pch=20, main="t-SNE", xlab="", ylab="", col=res.pam$clustering)
points(tsne_dist$Y[res.pam$id.med,], pch=4, col=1:6)
```



Die Struktur des Clustering ist auch in den Visualisierungen gut zu erkennen.

d) Erzeugen Sie eine Konfusionsmatrix zwischen dem Clustern und den Fraktionen `table(NR_meta$Fraktion, res.pam$clustering)`. Passen die Cluster zu den Fraktionen?

```
table(NR_meta$Fraktion, res.pam$clustering)
```

```
     1  2  3  4  5  6
BD   0  0  0  0  0  7
C    0  0 28  0  2  0
G    0 12  0  0  0  0
GL   0  0  0  0  8  0
RL   0  0  0 33  0  0
S    0 41  0  0  0  0
V   68  0  0  0  0  0
```

Die Cluster decken die Fraktion sehr gut ab, wobei die Grünen und die SP sowie die Grüne Liberalen und die CVP in einem Cluster sind.

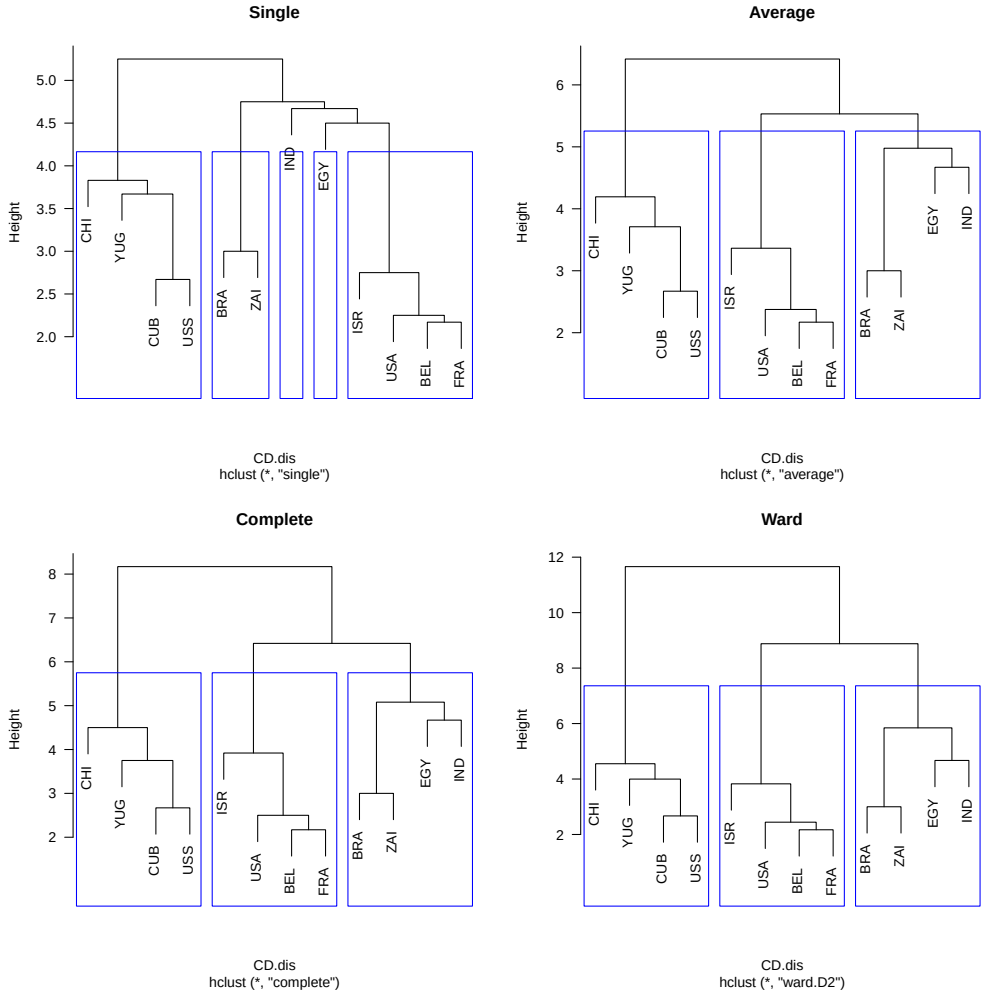
Aufgabe 4: Hierarchisches Clustering

Die Matrix `CD.dis` im File `CountriesDis.RDA` (mit `load()` laden) enthält Unähnlichkeiten zwischen Ländern. Die Daten stammen aus einer etwas älteren Studie, in der Studierende aufgefordert waren, paarweise die Unähnlichkeit zwischen den 12 Ländern Belgien (BEL), Brasilien (BRA), Chile (CHI), Kuba (CUB), Ägypten (EGY), Frankreich (FRA), Indien (IND), Israel (ISR), Vereinigte Staaten (USA), Sowjetunion (USS), Jugoslawien (YUG) und Zaire (ZAI) anzugeben. In der Unähnlichkeitsmatrix sind die durchschnittlichen Bewertungen der Studierenden festgehalten.

a) Führen Sie mit dieser Unähnlichkeitsmatrix hierarchische Cluster-Analysen durch, verwenden Sie verschiedene Linkage-Methoden. Vergleichen Sie dabei die Resultate der Cluster-Methoden `single`, `complete`, `average` und `ward.D2` bezüglich der Gruppenbildung.

```
load('Daten/CountriesDis.RDA')
c.single <-  hclust(CD.dis, method="single")
c.average <-  hclust(CD.dis, method="average")
c.complete <-  hclust(CD.dis, method="complete")
c.ward <-  hclust(CD.dis, method="ward.D2")
par(mfrow=c(2,2))
plot(c.single, labels=labels(CD.dis), main="Single", las=1)
```

```
rect.hclust(c.single, h=4, border="blue")
plot(c.average, labels=labels(CD.dis), main="Average", las=1)
rect.hclust(c.average, h=5, border="blue")
plot(c.complete, labels=labels(CD.dis), main="Complete", las=1)
rect.hclust(c.complete, h=5.5, border="blue")
plot(c.ward, labels=labels(CD.dis), main="Ward", las=1)
rect.hclust(c.ward, h=8, border="blue")
```



Clusterbildung bei

Single L.: Die Lücke ist bei 3.9 bis 4.4. Schneidet man bei 4, so erhalten wir drei Cluster und 2 Mavericks (IND, EGY). Ein Cluster besteht aus CHI, Yug, Cub, USS (kommunistische Länder), ein zweiter aus BRA, ZAI und ein dritter aus ISR, USA, BEL, FRA (westliche Länder)

Average L.: Hier ist nicht so klar, wo man scheiden soll. Schneidet man bei 5 gibt, führt das zu drei Clustern: Kommunistische Länder, westliche Länder und die restlichen Länder.

Complete L.: Hier schneidet man am besten bei 5.5, weil die Lücke verhältnismässig gross ist. Das führt zu drei Clustern. Es sind die identischen Cluster wie beim Average Linkage.

Ward L.: Hier kann man klar bei 8 schneiden, sodass es drei Cluster gibt. Die Cluster sind identisch

mit der Average bzw. Complete Linkage Methode.

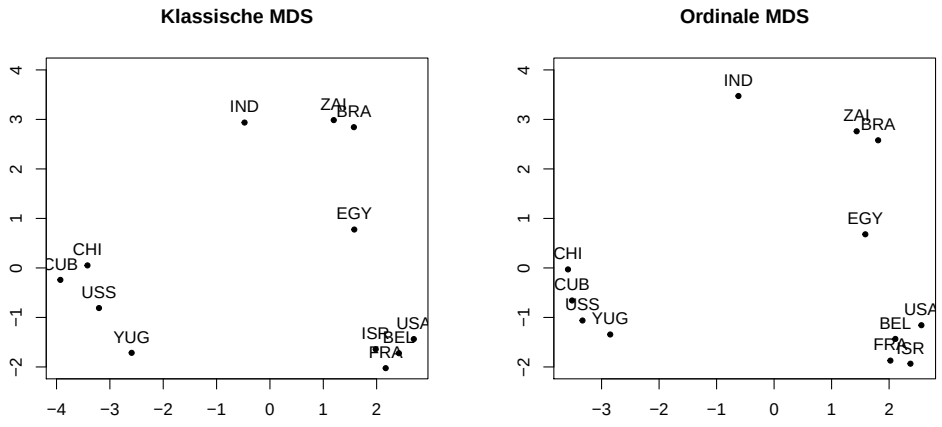
Die Cluster-Bildung durch die vier Methoden ergibt ein fast einheitliches Bild. Bei allen Methoden sehen wir die zwei Cluster "kommunistische Länder" und "westliche Länder". Ob die restlichen Länder einen Cluster bilden oder in zwei zerfallen, hängt jedoch von der verwendeten Methode ab.

b) Inwiefern finden Sie in einer multidimensionalen Skalierung die Resultate der hierarchischen Cluster-Analyse wieder?

```
par(mfrow=c(1,2))
## Klassische MDS
res.cmd <- cmdscale(CD.dis)
plot(res.cmd, pch=20, main="Klassische MDS", xlab="", ylab="", ylim=c(-2,4))
text(res.cmd, labels=rownames(res.cmd), pos=3)
## Orindale MDS
library("MASS")
res.iso <- isoMDS(CD.dis)
```

```
initial  value 11.171833
iter   5 value 8.475005
iter   5 value 8.471784
iter   5 value 8.470026
final  value 8.470026
converged
```

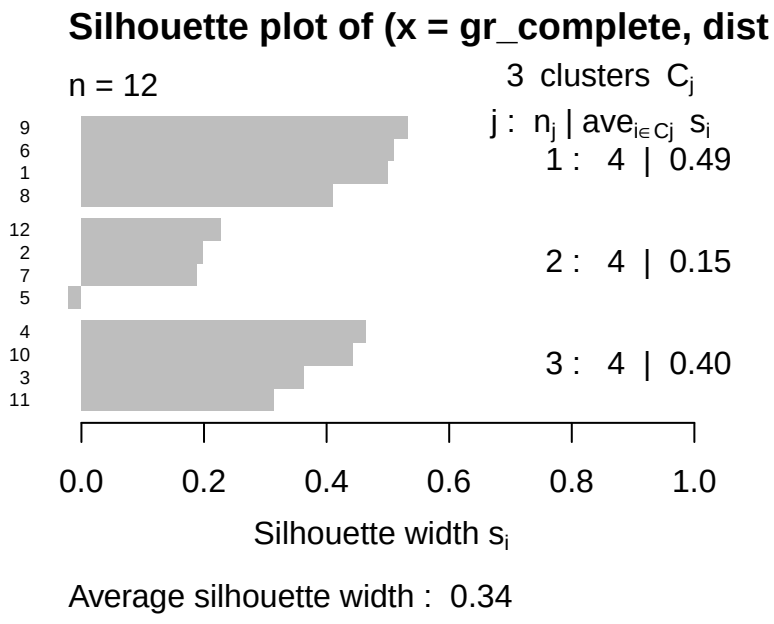
```
plot(res.iso$points, pch=20, main="Ordinale MDS", xlab="", ylab="", ylim=c(-2,4))
text(res.iso$points, labels=rownames(res.cmd), pos=3)
```



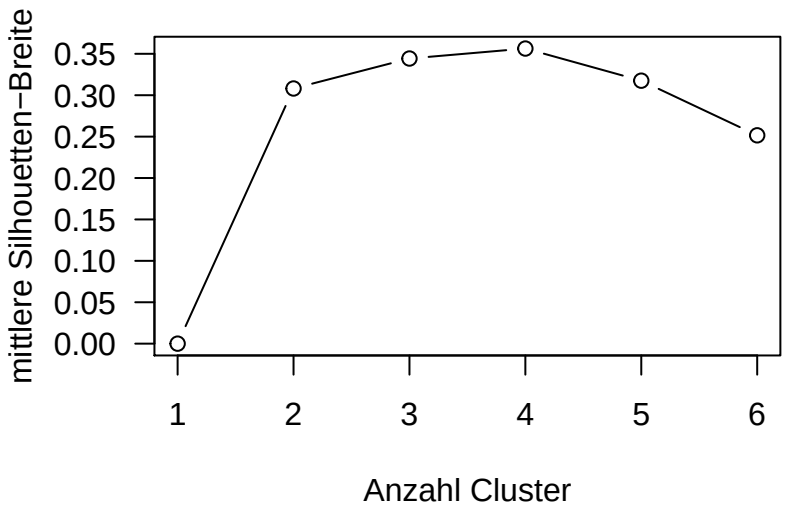
Da die Daten einem Ranking entstammen, wäre hier eine orindale MDS besser geeignet. Die Unterschiede zum klassischen Ansatz sind aber gering. Die Punkte liegen auf zwei "Bananen". In der linken sind die vier Punkte sehr regelmässig verteilt. Es handelt sich um die kommunistischen Länder. In der rechten Banane hat es weiter Unterstrukturen. Unten liegen die westlichen Länder, dann folgt gegen oben EGY, dann das Duo BRA und ZAI und am oberen Ende liegt IND.

c) Analysieren Sie das Ergebnis des hierarchischen Clusters mit dem Complete-Linkage Ansatz mit dem Silhouetten-Plot. Um aus dem Dendogramm die Cluster für eine spezifische Anzahl Cluster oder eine spezifische Höhe zu erhalten, können Sie die R-Funktion `cutree` verwenden. Können Sie mit der Silhouetten-Methode auch die optimale Anzahl Cluster bestimmen.

```
gr_complete <-  cutree(c.complete, h=5.5)
plot(silhouette(x=gr_complete, dist=CD.dis), cex.names=0.6)
```



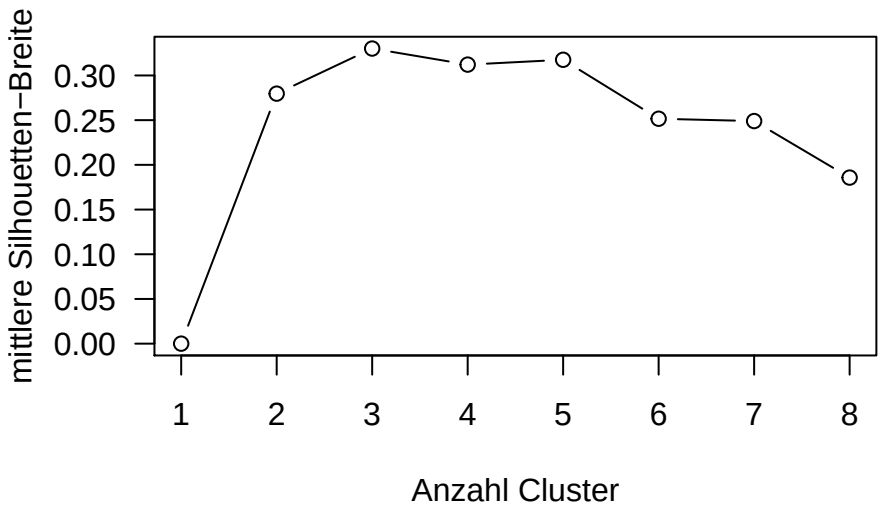

```
sil <-  rep(0, 6) # Initializierung
for (i in 2:6) sil[i] <- summary(silhouette(x=cutree(c.complete, k=i),
                                            dist=CD.dis))$avg.width
plot(x=1:6, sil, type = "b", xlab = "Anzahl Cluster",
     ylab = "mittlere Silhouetten-Breite", las=1)
```



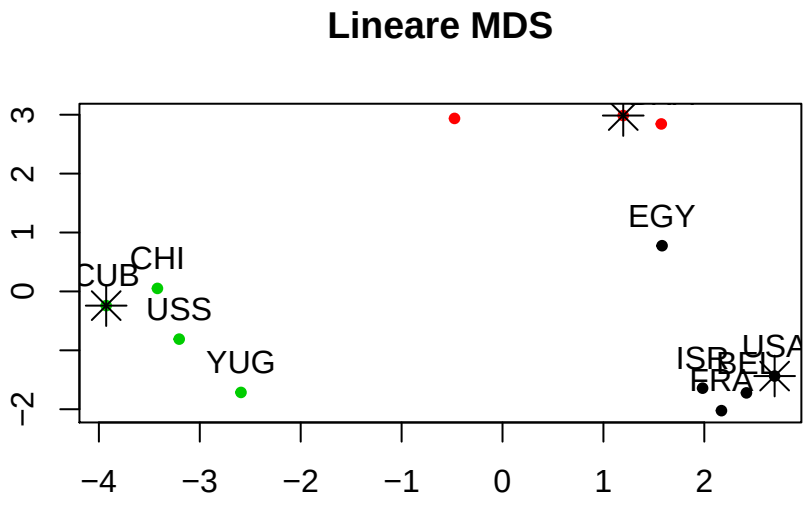
Die optimale Anzahl Klassen beträgt mit diesem Ansatz 4. Die Daten zeigen aber insgesamt nur eine schwache Struktur.

d) Führen Sie das Clustern auch noch mit einem Partitionsverfahren (K-means oder PAM) durch. Bestimmen Sie die optimale Anzahl Cluster und visualisieren Sie das Ergebnis. Welches sind die typischen Vertreter der einzelnen Cluster?

```
# Anzahl Cluster
sil <-  rep(0, 8) # Initializierung
for (i in 2:8) sil[i] <- summary(silhouette(x=pam(CD.dis, k=i)$cluster, dist=CD.dis))$avg.width
plot(x=1:8, sil, type = "b", xlab = "Anzahl Cluster", ylab = "mittlere Silhouetten-Breite", las=1)
```




```r
# 3 Cluster
# PAM
set.seed(7)
res.pam <- pam(CD.dis, k=3)
#Visulisierung
res.cmd <- cmdscale(CD.dis)
plot(res.cmd, pch=20, main="Lineare MDS", col=res.pam$cluster, xlab="", ylab="")
text(res.cmd, labels=rownames(res.cmd), pos=3)
points(res.cmd[rownames(res.cmd) %in% res.pam$medoids,],  pch = 8, cex = 2)
```

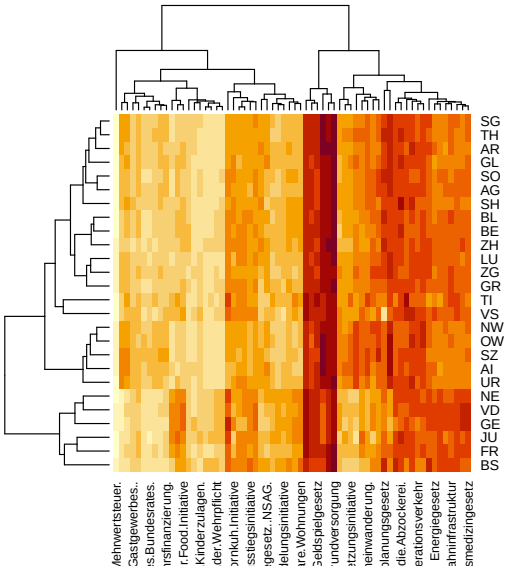


Sie können hier nur mit dem PAM-Algorithmus Clustern, für K-Means brauchen Sie die Distanzmatrix. Hier haben Sie nur die Distanzen.
Optimal sind auch hier drei Cluster. Ägypten wir hier allerdings den westlichen Länder zugewiesen.
Als typische Vertreter können die Medoids betrachte werden: USA, ZAI und CUB.

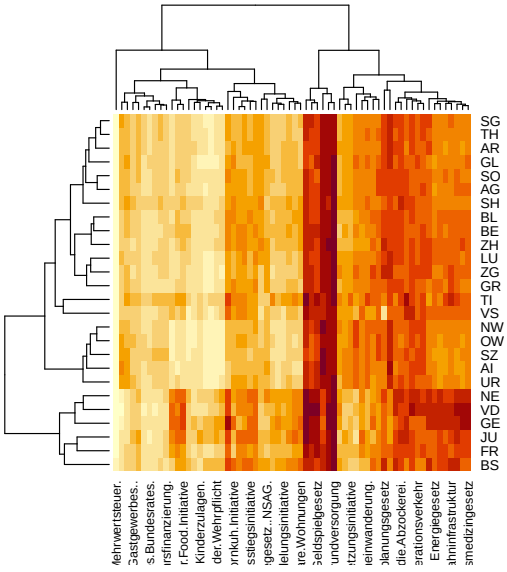
Aufgabe 5: Heatmap

Betrachten Sie noch einmal die Abstimmungsdaten und visualisieren Sie die Daten mit einer Heatmap (`heatmap`). `heatmap` erwartet als Input eine numerische Matrix. Sie müssen die Daten darum mit `as.matrix(...)` noch anpassen. Spielt es eine Rolle, wie Sie die Daten skalieren (Argument `scale`)? Wenn Sie Lust haben, können Sie auch die Alternative `pheatmap` aus dem Paket `pheatmap` testen.

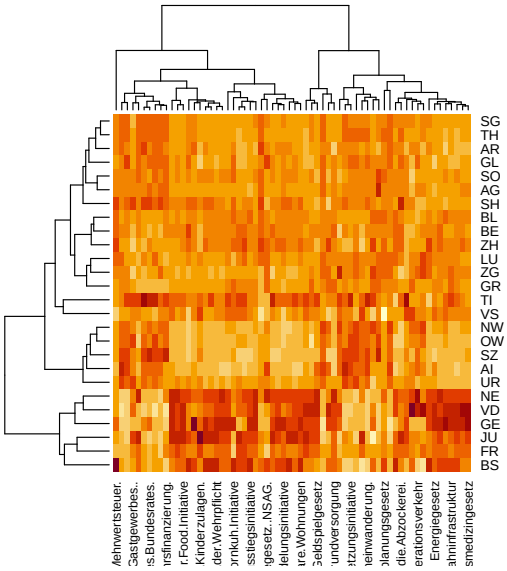
```
load("Daten/abst.Rdata")
## Standard
heatmap(as.matrix(abst), scale="row")
```




```
heatmap(as.matrix(abst), scale="none")
```




```
heatmap(as.matrix(abst), scale="column")
```




```
##Elegante Alternative
library(pheatmap)
pheatmap(abst, scale="none", cex=0.9)
```

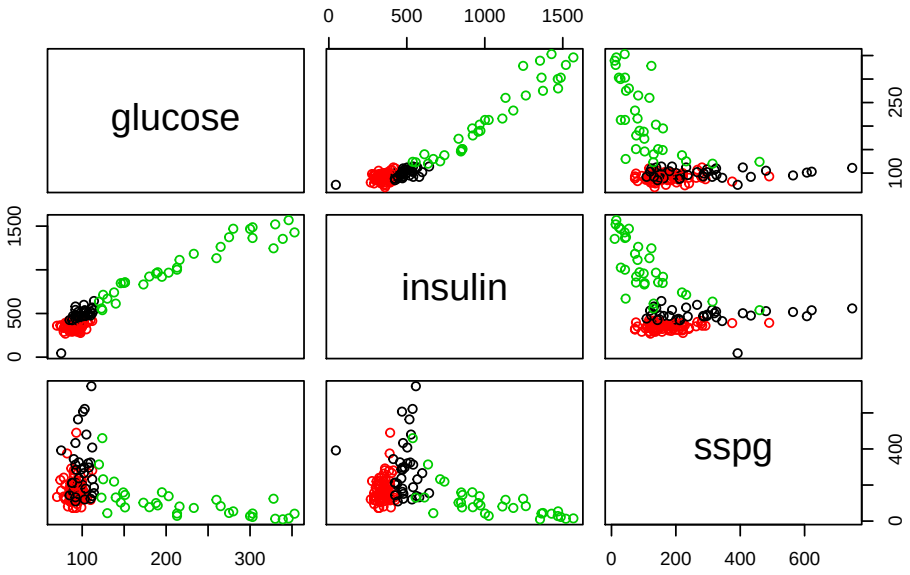
Da die Daten hier alle die gleiche Einheit aufweisen, braucht man hier nicht zu skalieren. Ohne Skalierung und mit Skalierung der Reihen erhält man jedoch ähnlich Resultate. Der Grund dafür ist, dass die Streuung in den einzelnen Kantonen ähnlich ist. Die Skalierung hat darum keinen grossen Einfluss.
In den Abbildungen sind verschiedene vertikale Streifen erkennbar. Diese Trennen verschiedenen Gruppen von Abstimmungen, Beim näheren Hinsehen sieht man auch noch zwei horizontale Gruppen (Lateinisch plus urbane Gegenden gegen ländliche Gegenden).
Standardisiert man die Spalten kann man nur noch Unterschiede zwischen den Kantonen erkennen.

Aufgabe 6: Diabetes

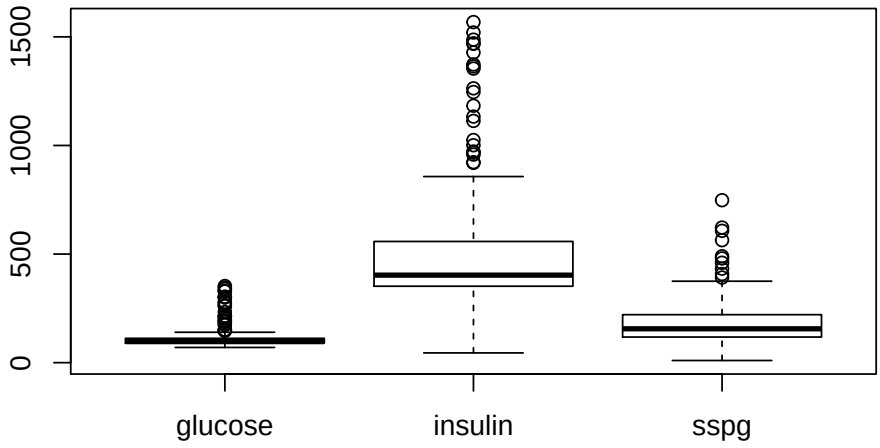
In dieser Aufgabe arbeiten wir mit dem Datensatz diabetes aus dem Paket `mclust`. Dazu müssen Sie zuerst das Paket installieren (`install.packages("{}mclust")` und laden (`library(mclust)`). Die Daten stehen Ihnen dann zur Verfügung. Informationen zu den Daten gibt es mit `?diabetes`.

a) Stellen Sie die Daten graphisch dar. Was sehen Sie? Macht es Sinn die Daten für das Clustering zu normalisieren?

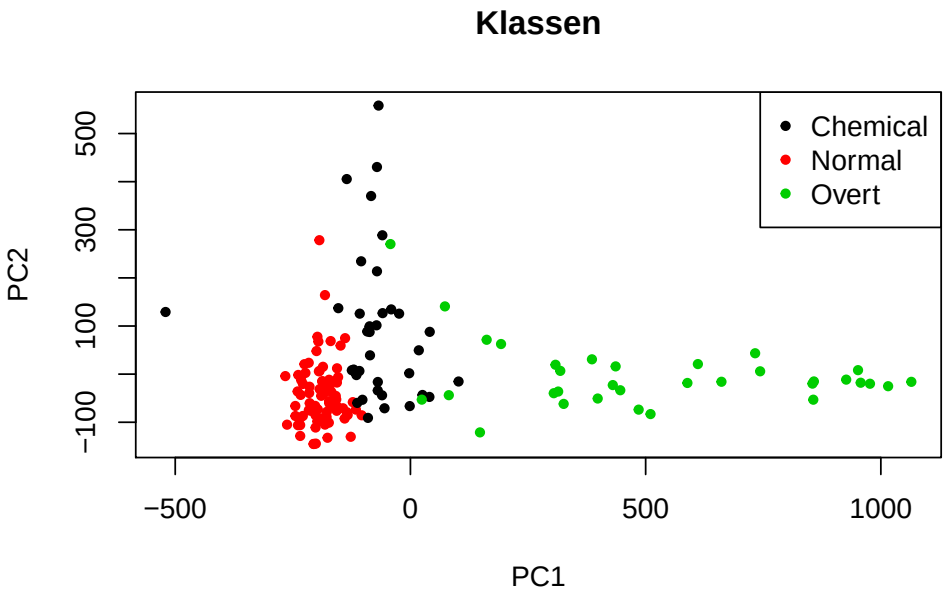
```
library(mclust)
data("diabetes")
pairs(diabetes[,-1],col=diabetes[,1])
```




```
boxplot(diabetes[,-1])
```



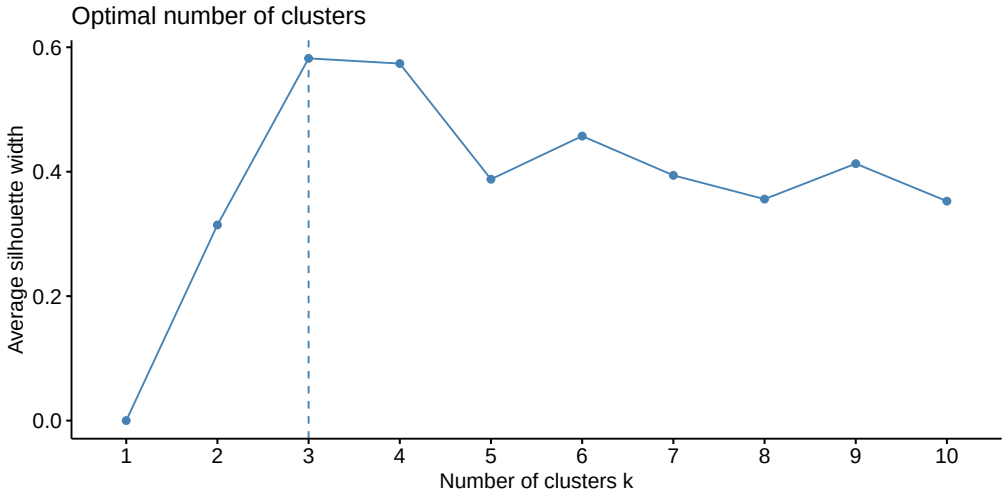

```
## PCA für visualisierung
res.pca <- prcomp(diabetes[,-1])
plot(res.pca$x, pch=20, main="Klassen", col=diabetes$class)
legend("topright", legend=levels(diabetes$class), col=1:3, pch=20)
```



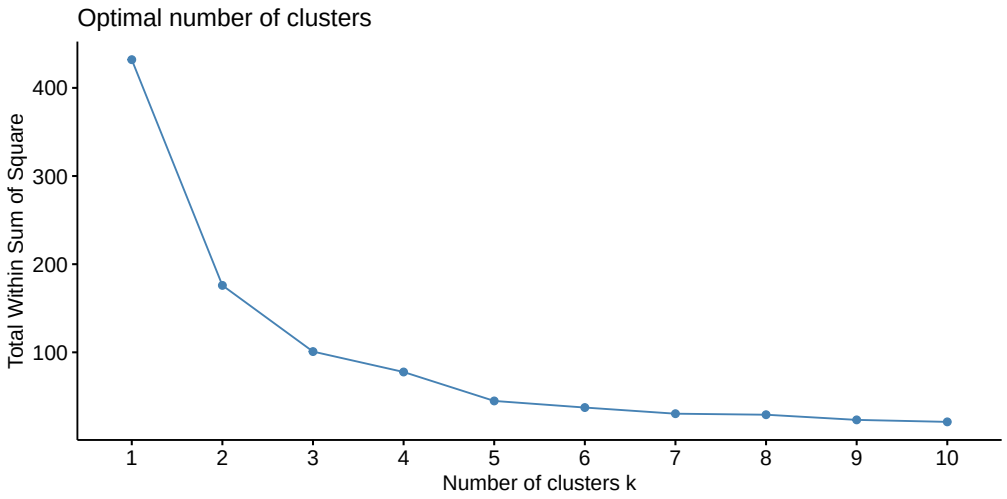
Die Daten sind teilweise korreliert. Im pairs-Plot kann man die verschiedenen Klassen nicht direkt identifizieren. Da die Daten auf unterschiedlichen Skalen sind, macht es Sinn die Daten zu skalieren.

b) Führen Sie mit einem herkömmlichen Ansatz (K-Means, pam oder hierarchisches Clustering) ein Clustering der standardisierten Variablen `glucose`, `insulin` und `sspg` durch (`scale(diabetes[, -1])`. Vergleichen Sie Ihr Clusterergebnis mit den tatsächlichen Gruppen in der Variable `class` mit einer Konfusionsmatrix (`table(cluster=res$cluster, diabetes$class)`. Was sehen Sie?

```
diab_sd <- scale(diabetes[, -1])
## K-Means
par(mfrow=c(1,2))
library(factoextra)
fviz_nbclust(diab_sd, kmeans, method="silhouette")
```



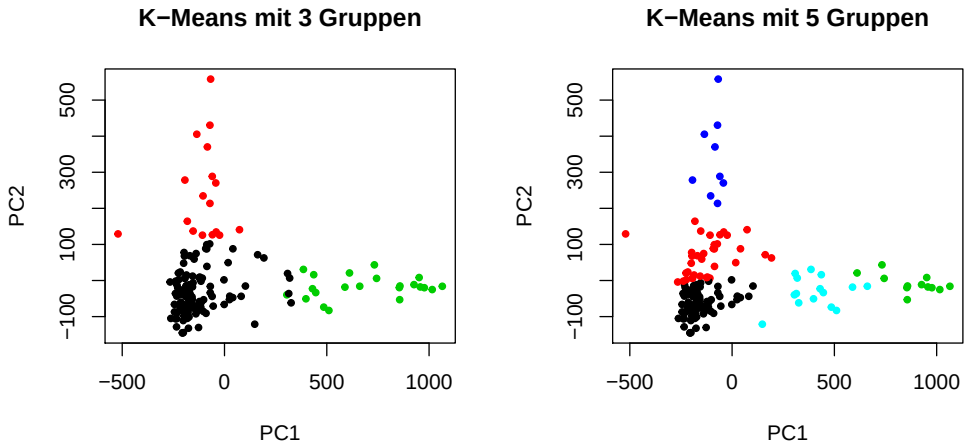
```
fviz_nbclust(diab_sd, kmeans, method="wss")
```




```
res.kmeans3 <- kmeans(diab_sd, centers=3)
plot(res.pca$x, pch=20, main="K-Means mit 3 Gruppen", col=res.kmeans3$cluster)
table(Cluster=res.kmeans3$cluster, diabetes$class)
```

```
Cluster Chemical Normal Overt
      1       23     74     9
      2       13      2     2
      3        0      0    22
```

```
res.kmeans5 <- kmeans(diab_sd, centers=5)
plot(res.pca$x, pch=20, main="K-Means mit 5 Gruppen", col=res.kmeans5$cluster)
```

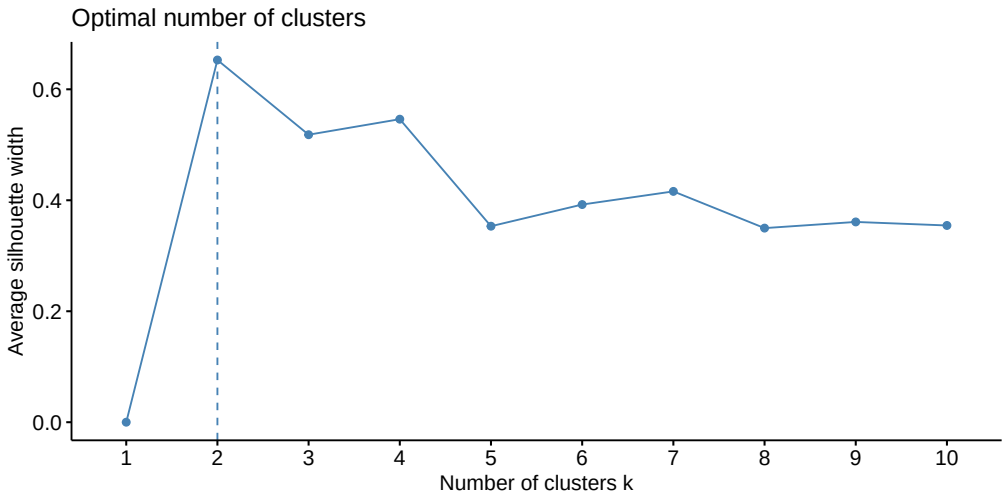



```
table(Cluster=res.kmeans5$cluster, diabetes$class)
```

```
Cluster Chemical Normal Overt
      1       13     59     2
      2       16     16     3
```

```
      3         0       0     12
      4         7       1      1
      5         0       0     15
```

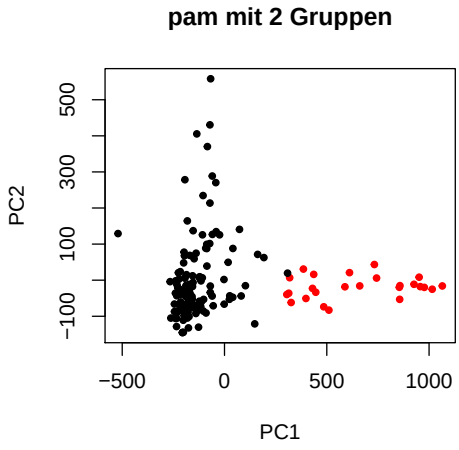
```
## PAM
library(cluster)
fviz_nbclust(diab_sd, pam, method="silhouette")
```

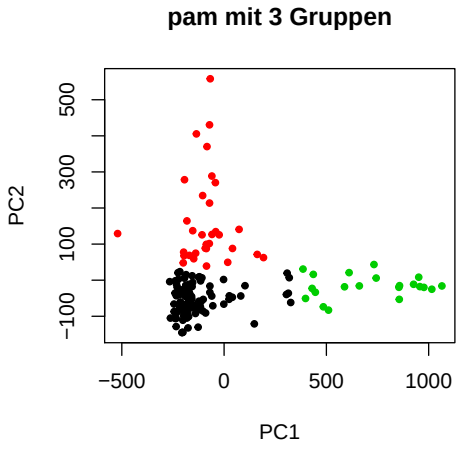



```
res.pam2 <- pam(diab_sd, k=2)
plot(res.pca$x, pch=20, main="pam mit 2 Gruppen", col=res.pam2$clustering)
table(Cluster=res.pam2$clustering, diabetes$class)
```

```
Cluster Chemical Normal Overt
      1       36     76     8
      2        0      0    25
```

```
res.pam3 <- pam(diab_sd, k=3)
plot(res.pca$x, pch=20, main="pam mit 3 Gruppen", col=res.pam3$clustering)
```

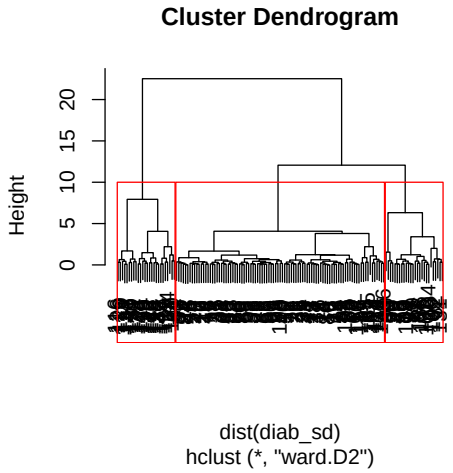


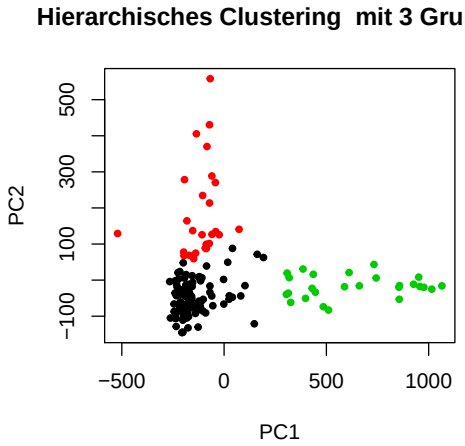


```
table(Cluster=res.pam3$clustering, diabetes$class)
```

```
Cluster Chemical Normal Overt
      1       16     68     8
      2       20      8     4
      3        0      0    21
```

```
## Hierarchisches Clustering
res.hc <-  hclust(dist(diab_sd), method="ward.D2")
plot(res.hc)
rect.hclust(res.hc, k=3, border="red")
plot(res.pca$x, pch=20, main="Hierarchisches Clustering  mit 3 Gruppen", col=cutree(res.hc, k=3))
```





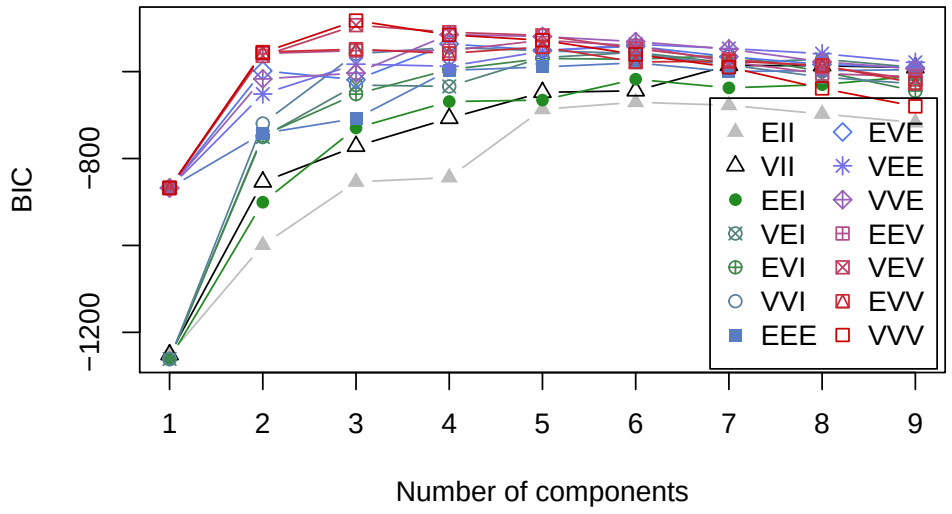

```
table(Cluster=cutree(res.hc, k=3), diabetes$class)
```

```
Cluster Chemical Normal Overt
      1       19     69     5
      2       17      7     2
      3        0      0    26
```

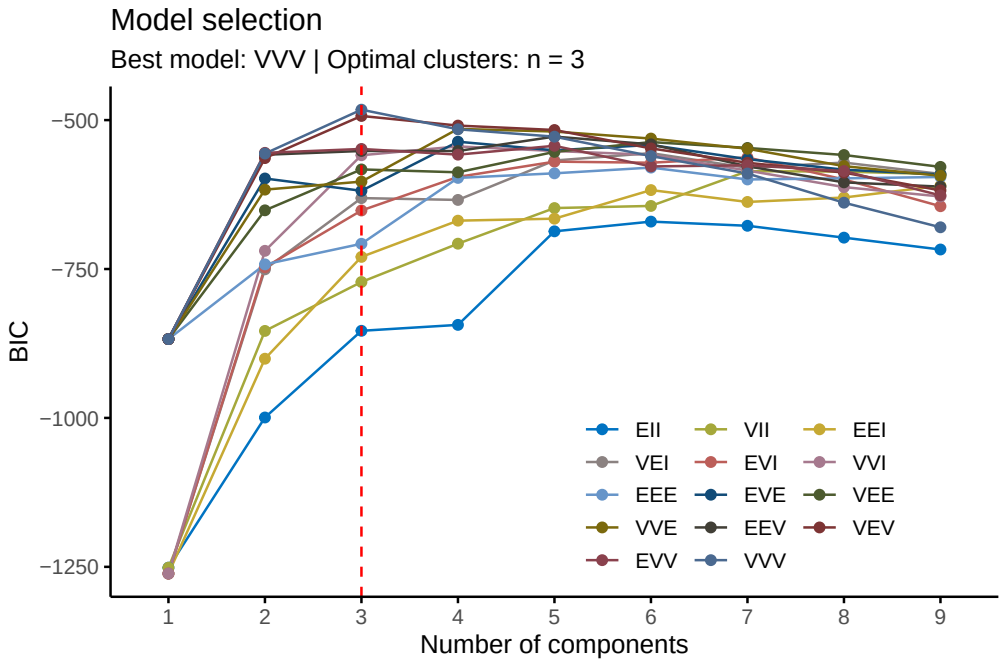
Die Ergebnisse der Ansätze sind ähnlich. Insbesondere die Gruppen Chemical und Normal werden nicht gut unterschieden. Je nach Ansatz werden auch nur 2 oder mehr als 3 Gruppen erkannt.

c) Führen Sie nun die Clusteranalyse mit dem modellbasierten Clustering durch.

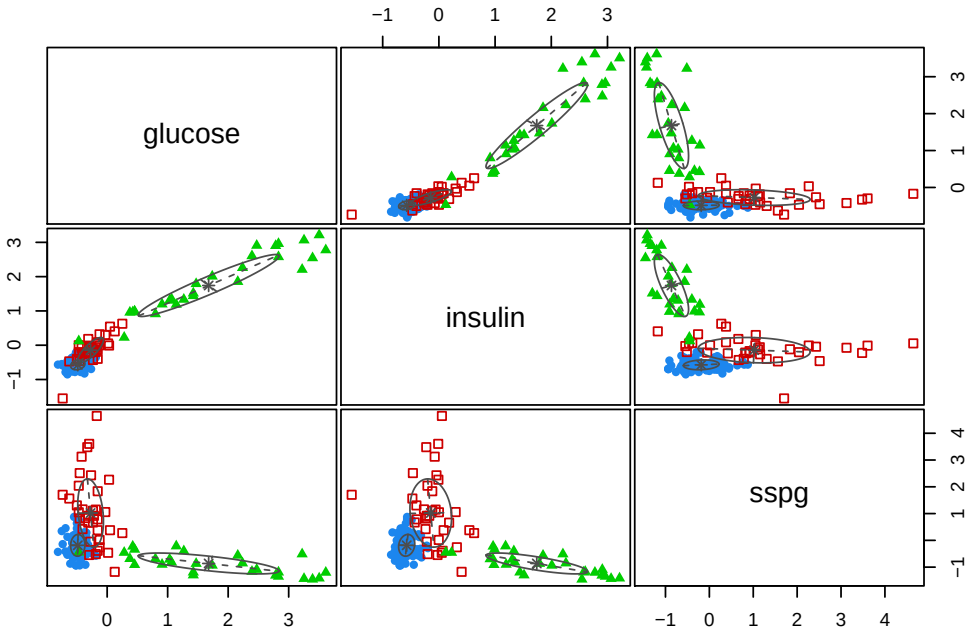
```
res.mc <- Mclust(diab_sd, verbose = FALSE) ## mit verbose FALSE wird der Progressbar unterdrückt
plot(res.mc, what="BIC")
```



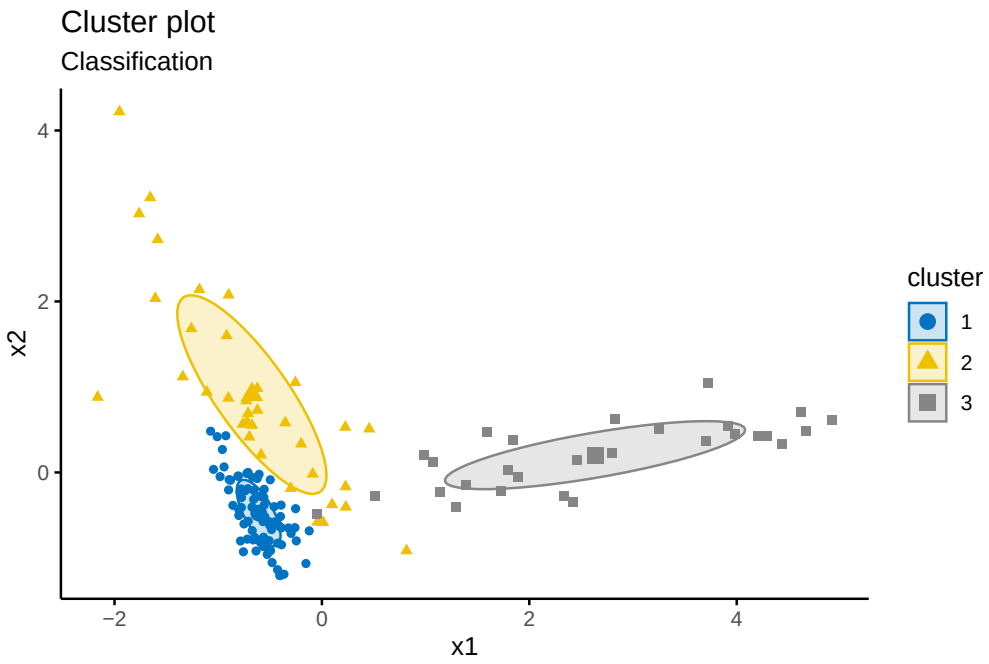

```
fviz_mclust(res.mc, "BIC",  palette = "jco")
```




```
plot(res.mc, what="classification")
```




```
fviz_mclust(res.mc, "classification", geom = "point",  palette = "jco", xlab="x1", ylab="x2")
```




```
table(Cluster=res.mc$classification, diabetes$class)
```

```
Cluster Chemical Normal Overt
      1        9     72     0
      2       26      4     6
      3        1      0    27
```

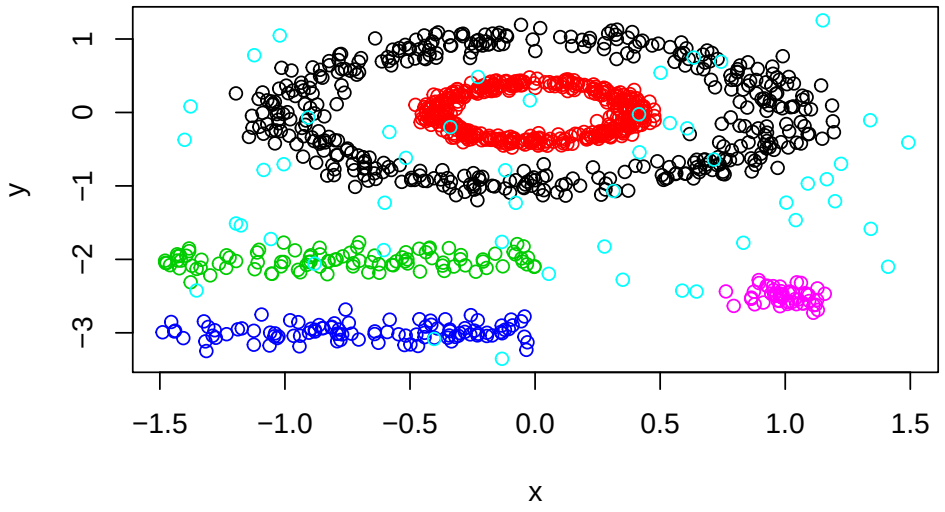
Mit dem modellbasierten Clustering gelingt es bei diesen Daten besser die Gruppen zu identifizieren. Zum Beispiel wird automatisch erkannt, dass es sich hier um 3 Gruppen handelt.

Aufgabe 7: Zusatz DBscan

In dieser Aufgabe wollen wir noch den DBscan Algorithmus für Dichteverbundenes Clustern anschauen. Dazu betrachten wir den Datensatz `multishapes` aus dem Paket `factorextra`. Das ist ein synthetischer Datensatz mit verschiedenen Formen, der generiert wurde, um die Fähigkeit von Dbscan zu demonstrieren. Die ersten beiden Spalten enthalten x- und y-Werte. Die dritte Spalte ist die Form Nummer.

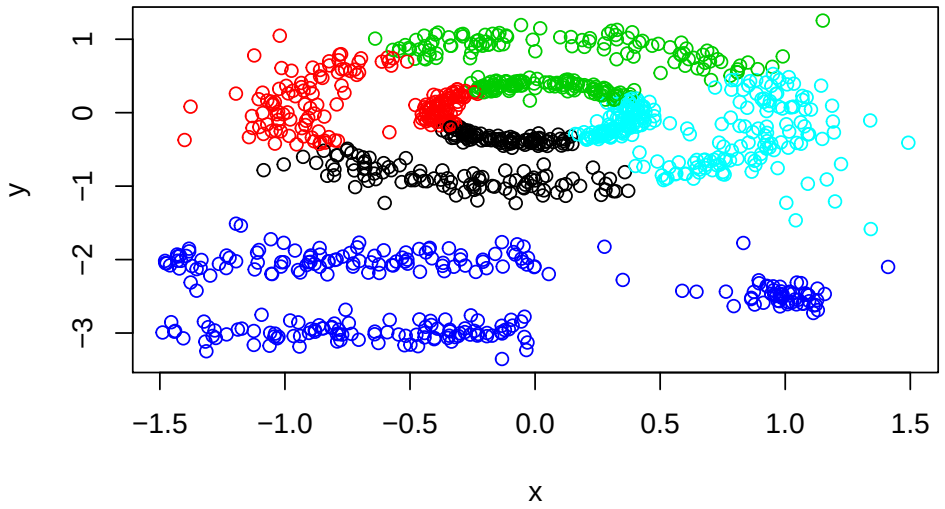
a) Visualisieren Sie den Datensatz.

```
library(factoextra)
data("multishapes")
plot(multishapes[,1:2], col=multishapes$shape)
```



b) Versuchen Sie die x- und y-Variablen mit K-Means in 5 Gruppen zu clustern.

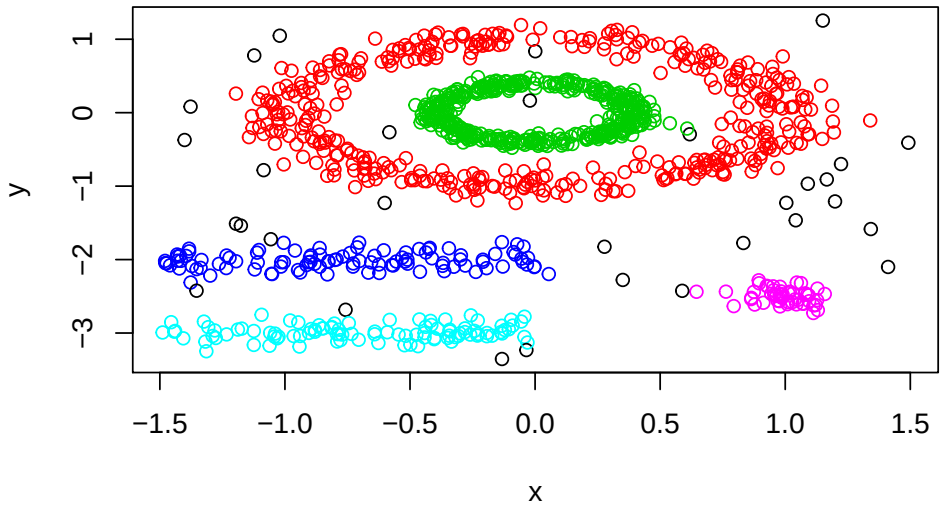
```
shape_kmeans <- kmeans(multishapes[,1:2], centers = 5)
plot(multishapes[,1:2], col=shape_kmeans$cluster)
```



Mit K-Means gelingt es nicht diese Formen zu identifizieren.

c) Nun versuchen wir die Daten mit DBSCAN zu clustern. Verwenden Sie dazu die Funktion dbscan aus dem Paket `dbscan` (muss vorgängig installiert werden). Neben den Daten müssen bei diesem Algorithmus die Argumente `minPts` (minimale Anzahl Punkte, die um einen Punkt liegen müssen, damit dieser ein Cluster wird) und `eps` (Umgebung welche für eps verwendet wird) definiert werden. Verwenden Sie hier die Werte `eps=0.15` und `minPts=5`. Die Clustereinteilung kann man aus dem Resultatobjekt mit `res$cluster` ziehen. Beachten Sie, dass Rauschpunkte mit 0 codiert sind. Will man die auch farblich darstellen, dann muss man in der plot-Funktion `col= res$cluster+1` schreiben.

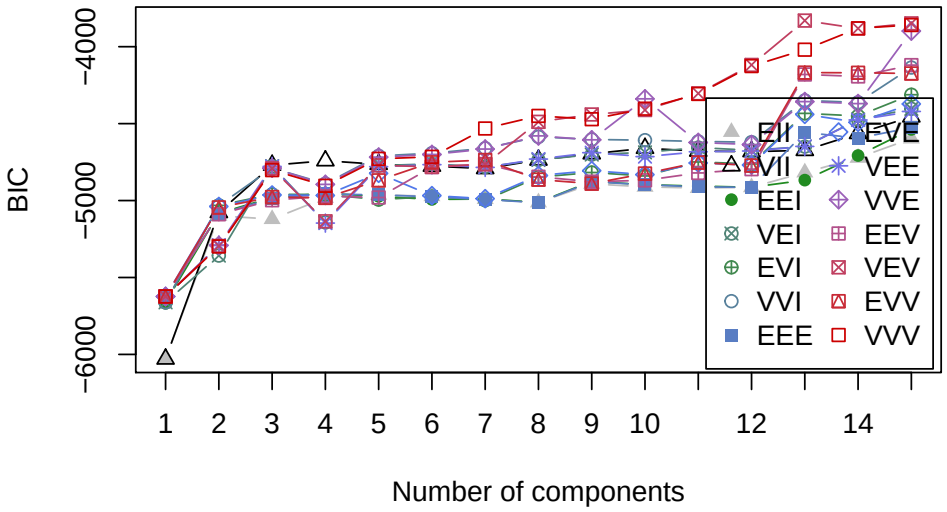
```
library(dbscan)
res.dbscan <- dbscan::dbscan(multishapes[,1:2], eps = 0.15, minPts = 5)
plot(multishapes[,1:2], col=res.dbscan$cluster+1)
```



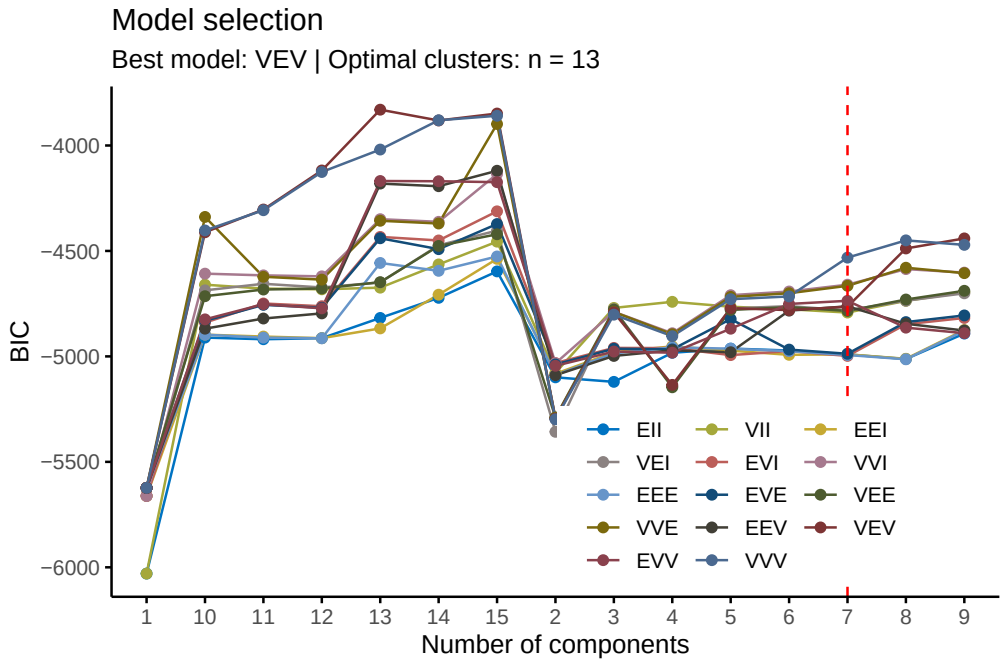
DBscan funktioniert hier sehr gut, wobei man natürlich auch bedenken muss, dass dieser spezifische Datensatz speziell auf DBscan ausgerichtet ist.

d) Nun können Sie auch noch versuchen die Daten mit einem modellbasierten Clustering zu identifizieren. Standardmässig werden in der Funktion `mclust` 1 bis 9 Gruppen gesucht. Das ist hier allenfalls nicht ausreichend. Um mehr Gruppen zu suchen, müssen Sie das Argument `G` anpassen.

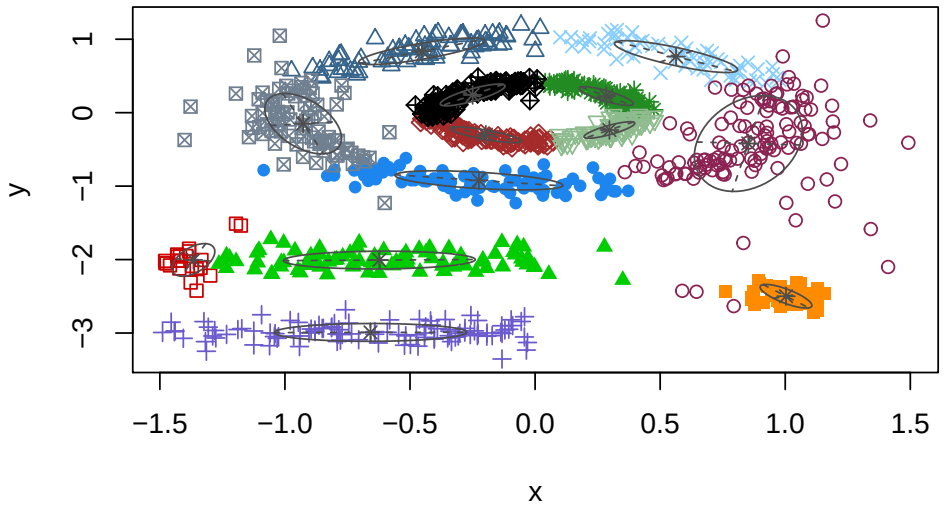
```
library(mclust)
mc <- Mclust(multishapes[,1:2], G=1:15, verbose = FALSE)
plot(mc, what="BIC")
```



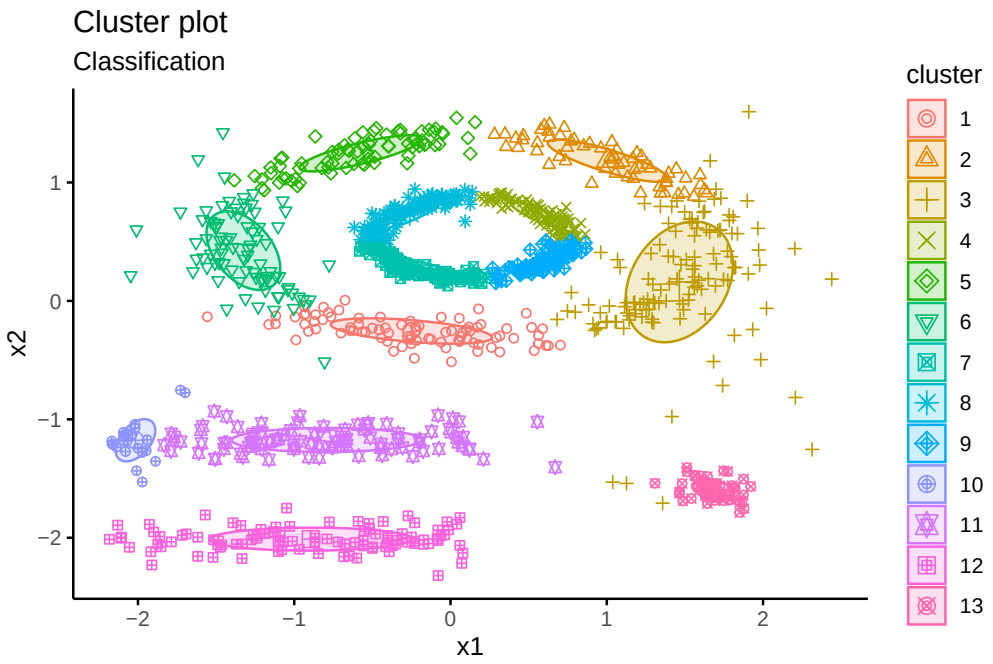

```
fviz_mclust(mc, "BIC",  palette = "jco")
```




```
plot(mc, what="classification")
```




```
fviz_mclust(mc, "classification", geom = "point",  xlab="x1", ylab="x2")
```



Modellbasiertes Clustern findet zu viele Gruppen. Kann aber immerhin die Formen unterscheiden.

Aufgabe 8: Leistungsnachweis

- Versuchen Sie Ihren Datensatz (oder Teile davon) mit einer geeigneten Methode zu clustern.
- Geben Sie an, wie Sie die Anzahl Cluster festlegen.

- Visualisieren Sie Ihre Cluster.
- Interpretieren Sie Ihre Ergebnisse.