Leistungsnachweis Modul C - Vollversion

Alexander Furrer

8. Juni 2020

Datensatz

Kurze Beschreibung des Datensatz

Der Datensatz enthält für 189 Länder 25 Attribute, welche aus öffentlich zugänglichen Quellen (IWF, SIPRI, Worldometers.info, UN, CIA) von mir zu einem Datensatz zusammengesetzt wurden. Der Datensatz umfasst insbesondere die folgenden Dimensionen:

Numerische Variablen

- Grösse der Bevölkerung (2016/2020)
- GDP absolut, pro Kopf und kaufkraftbereinigt
- Militärausgaben absolut und pro Kopf
- Covid19 Daten (Fälle, Tote, Test) absolut und pro 1M Bevölkerung

Kategoriale Variablen

- Länderidentifikation (Nummer, Code und Name)
- Politisches System (incl. Unabhängigkeit, Legigimationsbasis und Präsidentengewalt)
- Region und Subregion des Landes
- Hauptreligion im Land

Über die Analyse dieses Datensatzes sollen erstens die Anwendung der Methoden zur Dimensionsreduktions demonstriert werden, zweitens soll über Clustering versucht werden Muster in diesen Daten aufzufinden und drittens soll auf der Basis des Datensatzes eine Zielvariable (Region) via Klassifikation vorausgesagt werden. Ob dies mit dieser reichlich arbiträren Zusammenstellung von Variablen gelingt, wird sich noch weisen müssen.

Daten und Libraries laden

Der Datensatz wird nun zuerst eingelesen und zudem werden hier auch gleich alle benötigten Libraries geladen.

Kurze summary-Statistik

Die Daten können mit folgenden Befehlen kurz angeschaut werden. Dabei sieht man zum Beispiel, dass es in den Daten NA Werte hat. Zudem haben verschiedene numerische

Variablen auch Werte von 0. Auf das Andrucken wird hier aber verzichtet (das braucht zuviel Platz).

```
countries <- as_tibble(countries)
dim(countries)
head(countries)
summary(countries)
```

Die Übersicht der Struktur der Daten gibt zudem noch Hinweise auf eine benötigte Datenaufbereitung.

```
str(countries)
```

```
## 'data.frame':    189 obs. of  25 variables:
##  $ ISONumber                 : num  704 104 418 702 158 764 144 496 116 630 ...
##  $ ISOCode                   : chr  "VN" "MM" "LA" "SG" ...
##  $ ISOname                   : chr  " Viet Nam" " Myanmar" " Lao People's Democratic Republ
ic" " Singapore" ...
##  $ Population2016(t.)        : num  92691 52254 6585 5607 23540 ...
##  $ Population2020            : num  97296384 54392661 7270167 5848152 23814858 ...
##  $ GDP(nominalM$)            : num  201309 64366 15768 296966 529575 ...
##  $ GDP(nominalPC$)           : num  2172 1232 2394 52961 22497 ...
##  $ GDP(PPPM$)                : num  595368 303279 45246 492502 1132720 ...
##  $ GDP(PPPPC$)               : num  6423 5804 6871 87832 48119 ...
##  $ PurchasingPowerParity(PPP): num  34 21 35 60 47 35 31 30 34 82 ...
##  $ MilitarySpend2016M$       : num  NA NA NA 10218 10596 ...
##  $ MilitarySpendPC2016       : num  NA NA NA 1822 450 ...
##  $ COVIDTotalCases           : num  334 261 19 40604 443 ...
##  $ TotCases1MPop             : num  3 5 3 6943 19 ...
##  $ TotalDeath                : num  0 6 0 26 7 58 11 0 0 NA ...
##  $ Deaths1Mpop               : num  0 0.1 0 4 0.3 0.8 0.5 0 0 NA ...
##  $ TotalTest                 : num  275000 45926 10146 488695 73751 ...
##  $ Tests1Mpop                : num  2826 844 1396 83564 3097 ...
##  $ IndepenendState           : chr  "Yes" "Yes" "Yes" "Yes" ...
##  $ PolitSystem               : chr  "Republic" "Republic" "Republic" "Republic" ...
##  $ Headofstate               : chr  "Executive" "Executive" "Executive" "Ceremonial" ...
##  $ Legitimacy                : chr  "Power constitutionally linked to a single political mo
vement" "Presidency is elected by legislature; ministry partially subject to parliamentary con
fidence" "Power constitutionally linked to a single political movement" "Ministry is subject t
o parliamentary confidence" ...
##  $ Region                    : chr  "Asia" "Asia" "Asia" "Asia" ...
##  $ Subregion                 : chr  "South-eastern Asia" "South-eastern Asia" "South-easter
n Asia" "South-eastern Asia" ...
##  $ MainReligion              : chr  "Buddhist" "Buddhist" "Buddhist" "Buddhist" ...
```

Datenaufbereitung

Der Datensatz muss insbesondere aus drei Gründen noch etwas aufbereitet werden:

1. Die Kolonnennamen enthalten Sonderzeichen welche bereinigt werden müssen (Klammern, Punkte, etc)
2. Die katgoriellen Variablen müssen als Faktoren definiert werden
3. Für die Darstellung der PCA in Aufgabe 1 hat sich die Definition einer Kolonne mit einem Farbencode pro Region als hilfreich herausgestellt, was auch gleich hier schon gemacht wird.

Die Behandlung von NA-Werten und die Transformation der numerischen Daten wird hingegen nicht bereits hier, sondern auf dem Daten-Subset in der jeweiligen Aufgabe gemacht.

```
colnames(countries) <- c("ISONumber", "ISOCode", "ISOname", "Pop2016T",
                         "Pop2020", "GDPnomM", "GDPnomPC", "GDPparM",
                         "GDPparPC", "Parity", "MilitSpend2016M",
                         "MilitSpendPC2016" ,"COVIDTotalCases",
                         "Covid1MPop", "CovidDead", "CovidDead1Mpop",
                         "CovidTest", "CovidTest1Mpop", "IndepState",
                         "PolitSystem", "Headofstate", "Legitimacy",
                         "Region", "Subregion", "MainReligion")
countries$ISONumber <- as.factor(countries$ISONumber)
countries$ISOCode <- as.factor(countries$ISOCode)
countries$ISOname <- as.factor(countries$ISOname)
countries$IndepState <- as.factor(countries$IndepState)
countries$PolitSystem <- as.factor(countries$PolitSystem)
countries$Headofstate <- as.factor(countries$Headofstate)
countries$Legitimacy <-  as.factor(countries$Legitimacy)
countries$Region <-  as.factor(countries$Region)
countries$Subregion <-  as.factor(countries$Subregion)
countries$MainReligion <-  as.factor(countries$MainReligion)

# Farbenschema definieren fuer Regionen
countries$col[countries$Region == "Asia"]= "orange"
countries$col[countries$Region == "Europe"]= "blue"
countries$col[countries$Region == "Americas"]= "red"
countries$col[countries$Region == "Africa"]= "black"
countries$col[countries$Region == "Oceania"]= "green"
```

Visualisierung des Datensatzes

Daten-Subset definieren

Als erstes wird ein Daten-Subset *df.countries* definiert, das die relevanten numerischen Variablen enthält, um eine PCA Analyse zu machen. Dazu werden die normierten Variablen mit "Pro-Kopf" Werten verwendet, um die Populationsgrösse aus diesen Variablen zu entfernen. Zugleich werden auf diesem Set auch alle "NA" Werte ausgeschlossen, was die Anzahl Beobachtungen auf 148 reduziert.

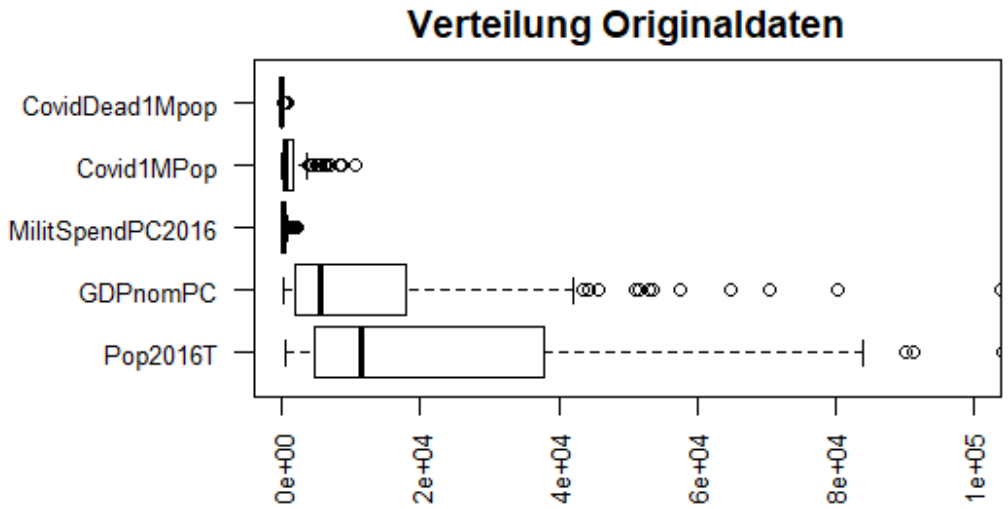
```
ohne.na <- complete.cases(countries[,c("Pop2016T", "GDPnomPC",
                                       "MilitSpendPC2016", "Covid1MPop",
                                       "CovidDead1Mpop")])
df.countries<- countries[ohne.na, c("Pop2016T", "GDPnomPC",
                                    "MilitSpendPC2016", "Covid1MPop",
                                    "CovidDead1Mpop", "ISOCode",
                                    "col", "PolitSystem", "Region",
                                    "MainReligion")]
```

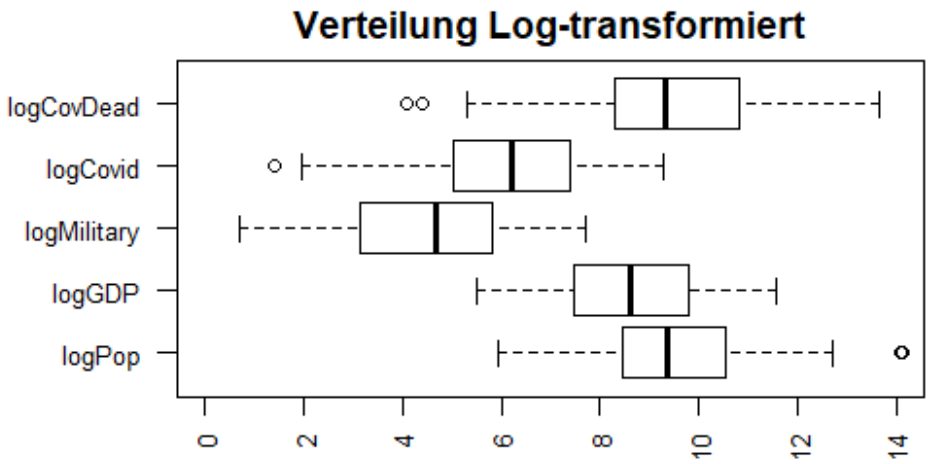
Um die rechtsschiefe Natur dieser Daten zu behandeln, wird eine Logtransformation angewendet. Das bedingt aber seinerseits den Ausschluss von ein paar wenigen Beobachtungen (14 Fälle) die Werte < 1 haben. Zudem werden die Covid-Todesfälle auf tausend Personen skaliert.

```
df.countries <- df.countries %>%
  filter(df.countries$MilitSpendPC2016 != 0,
         df.countries$CovidDead1Mpop != 0,
         df.countries$Covid1MPop > 0.9,)

df.countries$logPop <- log(df.countries$Pop2016T)
df.countries$logGDP <- log(df.countries$GDPnomPC)
df.countries$logMilitary <- log(df.countries$MilitSpendPC2016)
df.countries$logCovid <- log(df.countries$Covid1MPop)
df.countries$logCovDead <- log(df.countries$CovidDead1Mpop * 1000)
```

Nach der Transformation sehen die Werte dieser Variablen in der Boxplot-Darstellung ganz brauchbar aus und sollten für die PCA Analyse so verwendbar sein.

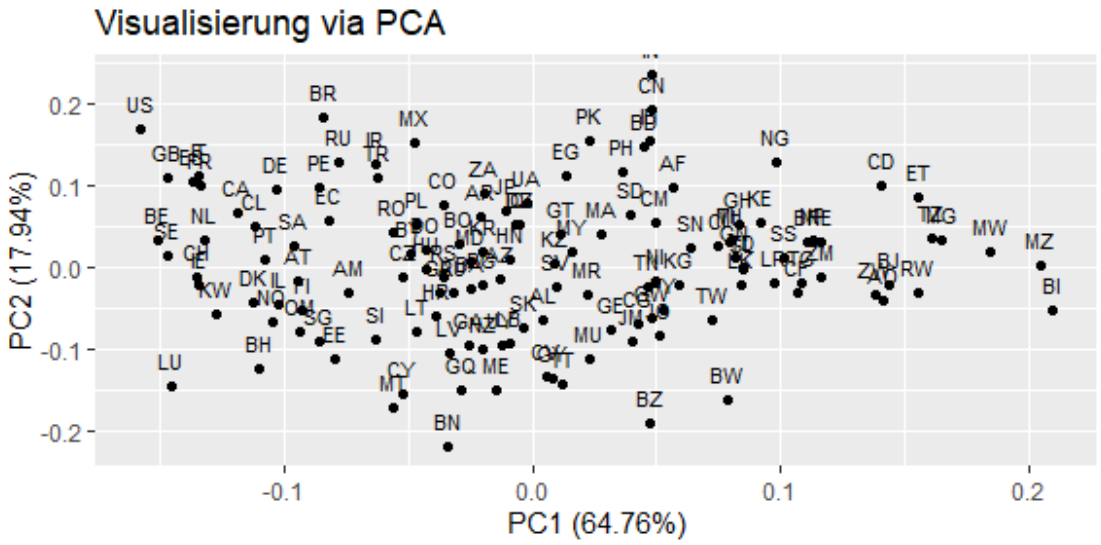




Visualisierung

Als nächsten Schritt wollen wir mit einer Hauptkomponenten-Analyse (PCA) den so aufbereiteten Datensatz visualisieren. Dazu verwenden wir die oben erstellten, logtransformierten Variablen (Kolonnen 11 bis 15) aus dem Daten-subset *df.countries* und verwenden die *autoplot*-Funktion zur Darstellung der ersten beiden Hauptkomponenten.

```
pca <- prcomp(df.countries[, c(11:15)], scale = FALSE)
autoplot(pca, main = "Visualisierung via PCA",
         cex = 0.8)+
  geom_text(aes(label=df.countries$ISOCode),
            hjust=0.5, vjust=-1, size = 3)
```

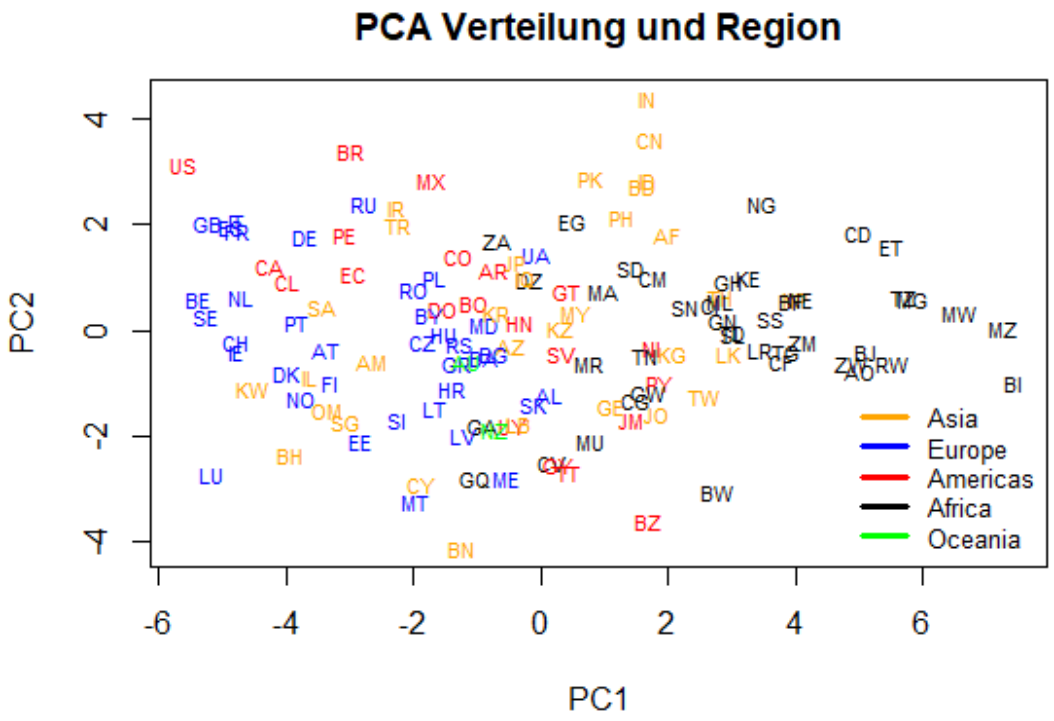


Mit diesen beiden Hauptkomponenten kann 82.7% der Streuung der Daten dargestellt werden (64.76% und 17.94%). Die Interpretation der Darstellung ist allerdings nicht

trivial, was man aber zum Beispiel sehen kann ist eine Gruppierung von reichen Staaten mit hohen Covid Werten (GB, IT, FR, ES). Die Rotationsmatrix der PCA zeigt, dass in der Komponente PC2 vor allem die Bevölkerungsgrösse eingeflossen ist (0.9273895), d.h. die grossen Länder sind in der Grafik eher oben angesiedelt. Eine zusätzliche Skalierung der Daten über das Argument *scale = TRUE* bringt aber keine grosse Veränderung der Zusammensetzung der Hauptkomponenten (PC1: 62.45%, PC2: 20.14%).

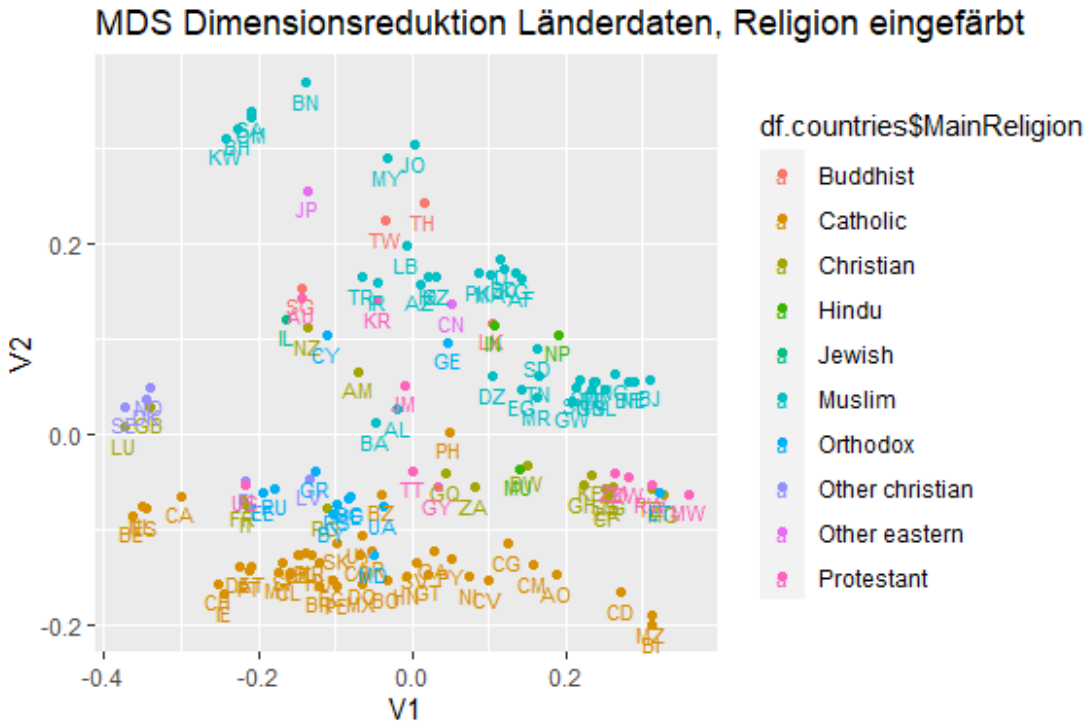
Eine interessante Ansicht ergibt sich allerdings wenn man die Länder zudem nach Region einfärbt. Hierbei sieht man, dass sich die afrikanischen Länder auf der rechten Seite der Grafik versammeln, Europa eher auf der linken Seite zu finden ist. Asien und Amerika verteilen sich "übers Kreuz" in der Mitte der Grafik, wobei US sich von Rest der Länder (Auch Americas) absetzt.

```
par(mfrow = c(1,1), mar = c(4, 4, 3, 2))
plot(pca$x[,1], pca$x[,2],                    # PC1 / PC2
     type="n",                                # no plotting
     xlab="PC1", ylab="PC2",                  # Achsenlabel
     main = "PCA Verteilung und Region")      # Titel
text(pca$x[,1], pca$x[,2],                    # x und y Positionierung
     labels = df.countries$ISOCode,           # Ländercodes anschreiben
     cex=0.7, col = df.countries$col)         # Font size und Farbe
legend('bottomright',
       legend = c('Asia', 'Europe', 'Americas', 'Africa', 'Oceania'),
       cex = 0.8, col=c('orange', 'blue', 'red', 'black', 'green'),
       lty = 1, lwd = 3, bty = "n")
```



Alternativ könnten die Daten auch über eine MDS unter Einbezug der Faktorvariablen aus *df.countries* angezeigt werden. Und zur Abwechslung sind die Länder hier nach Religion eingefärbt:

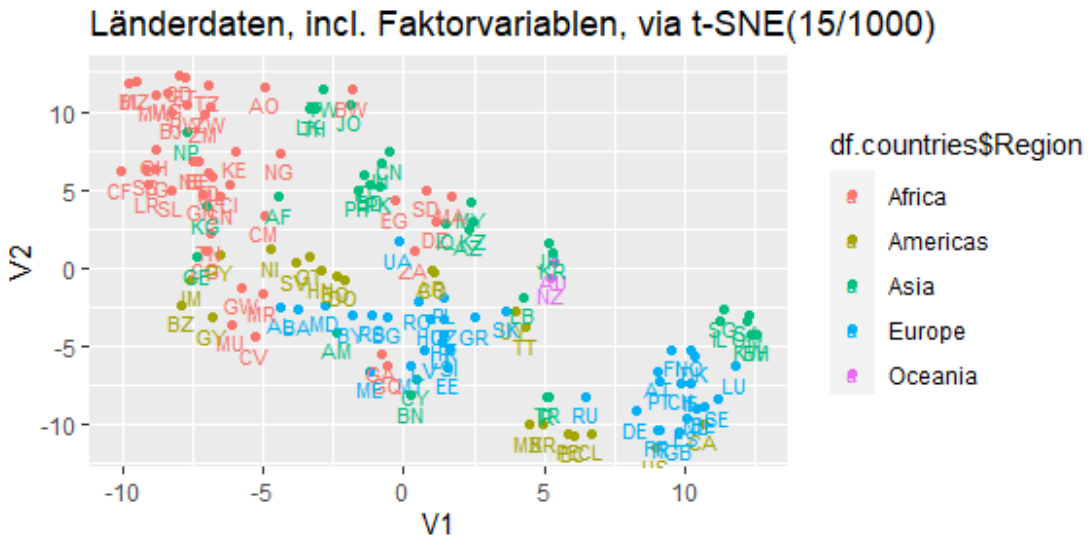
```
# Berechnung der Distanzmatrix
dist.countries <- daisy(df.countries[, c(8:15)],
                        metric = "gower",
                        type = list(ordratio=1, ordratio=2, ordratio=3))
dist.cmd <- cmdscale(dist.countries, k=2)         # Reduktion auf 2 Dimension
en
d_mds <- as.data.frame(dist.cmd)                  # Matrix convertieren
# Darstellung via MDS
ggplot(data = d_mds,
       aes(x = V1, y = V2, col = df.countries$MainReligion)) +
  geom_point() +
  geom_text(aes(label=df.countries$ISOCode),
            hjust=0.5, vjust=1.5, size = 3) +
  ggtitle("MDS Dimensionsreduktion Länderdaten, Religion eingefärbt")
```



Und schliesslich kann der Datensatz auch über die t-SNE Funktion ausgewertet und auf 2 Dimensionen dargestellt werden (hier sind die Farben wieder nach Region gewählt):

```
set.seed(7)
countries.tSNE2 <- Rtsne(df.countries[, c(8:15)],
                         perplexity = 15,
                         max_int = 1000,
                         metric = "gower",
                         type = list(ordratio=1))
```

```
d_tsne2 <- as.data.frame(countries.tSNE2$Y)
ggplot(data = d_tsne2,
       aes(x = V1, y = V2, col = df.countries$Region)) +
  geom_point() +
  geom_text(aes(label=df.countries$ISOCode),
            hjust=0.5, vjust=1.5, size = 3) +
  ggtitle("Länderdaten, incl. Faktorvariablen, via t-SNE(15/1000)")
```



Zur t-NSE Darstellung ist anzumerken, dass die *perplexity* nach verschiedenen Versuchen auf den relativ tiefen Wert von 15 eingestellt wurde, da eine höhere Zahl weniger Struktur in den Daten zeigt. Die Anzahl Wiederholungen (*max_iter) ist nach etwas ausprobieren auf dem Defaultwert von 1000 belassen worden.

FAZIT zur Dimensionsreduktion

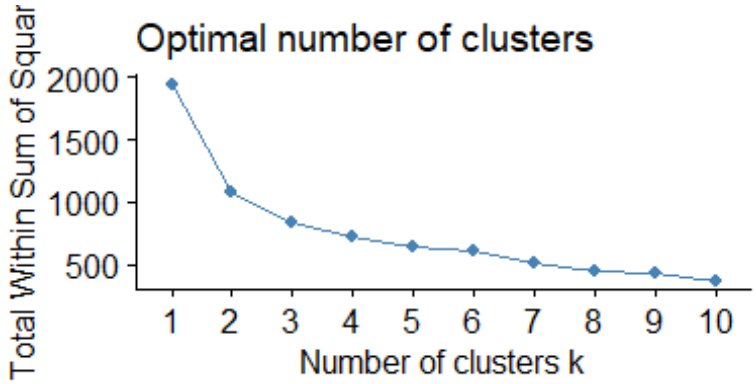
Die Dimensionsreduktion kann bei diesem Datensatz zwar technisch angewendet werden, die Daten führen aber zu keinen besonders intressanten Erkenntnissen. Hier rächt sich die komplett arbiträre Zusammensetzung des Datensatzes. Zudem führt wohl die Verwendung von zwei hochkorrelierten Variablen (Anzahl Covid Fälle und Anzahl Covid Tote) hier zu Schwierigkeiten. Man könnte das Ganze nun auch nochmals ohne die Variable *logCovid* machen und schauen ob die Resultate dann besser werden.

Clustering

Erster Versuch: kmeans auf der Datenmatrix *df.countires[11:15]*

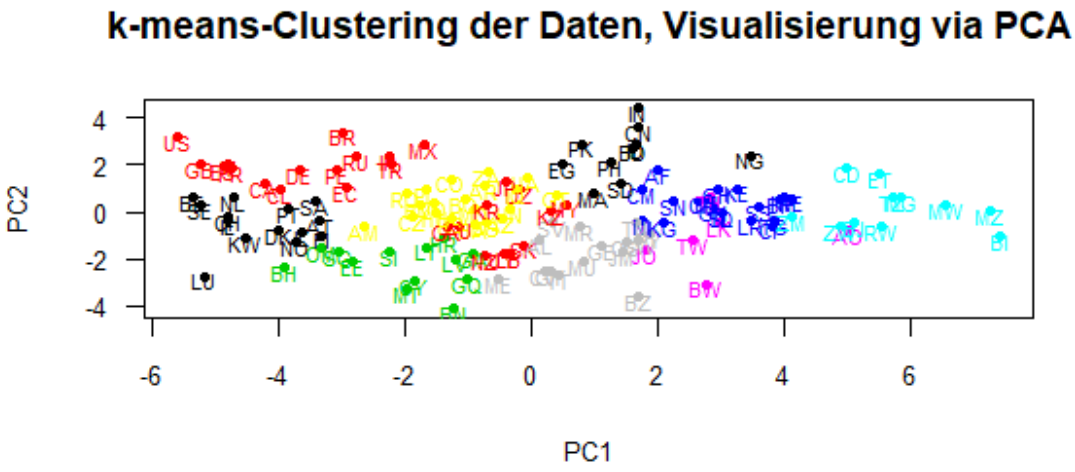
Als erster schneller Ansatz zum Clustern der Daten kann die K-Means-Methode angewandt werden, die die Varianz der euklidischen Distanzen innerhalb einer vorgegebenen Anzahl Cluster minimiert. Zur Bestimmung der optimalen Anzahl Cluster kann die Funktion *fviz_nbclust* aus der **factoextra**-Library verwendet werden.

```r
fviz_nbclust(df.countries[ ,11:15],
             kmeans, method="wss",
             k.max = 10)
```



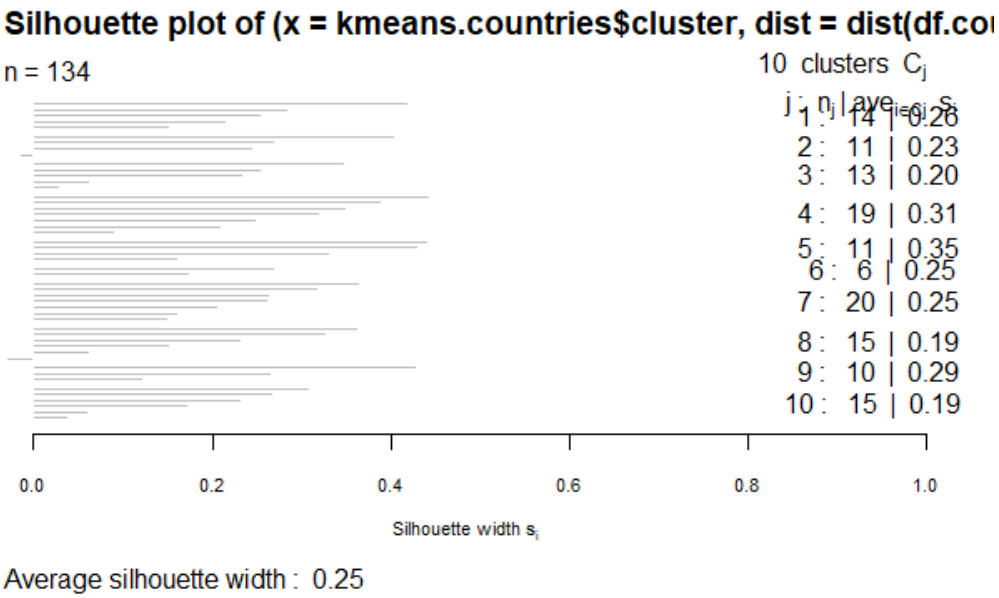
Das Resultat ist etwas enttäuschend, da sich aus der WSS Ananlyse kein spezifischer "Knick" ablesen lässt, der auf die optimale Anzahl Cluster hinweisen würde (und die wss-Werte fallen auch für *k > 10* weiter ab). Im weiteren wird hier aber einfach der beste Wert bei 10 Cluster gewählt.

```r
## Clustering
kmeans.countries <- kmeans(df.countries[,11:15], centers = 10)
## Visualisierung über PCA
pca2 <- pca$x[ , c(1, 2)]
plot(pca2, col = kmeans.countries$cluster,
     las = 1, pch = 20, cex.axis =0.8, cex.lab=0.8,
     main = "k-means-Clustering der Daten, Visualisierung via PCA")
text(pca$x[,1], pca$x[,2]-0.2,
     labels = df.countries$ISOCode,
     cex=0.7, col = kmeans.countries$cluster)
```



Die Darstellung zeigt 10 relativ klar voneinander abgegrenzte Cluster, wobei sich rein visuell insbeondere in der Mitte der Grafik einige Überlagerungen ergeben (die allerdings auch von der Dimensionsreduktion der PCA herkommen könnten). Die inhaltliche Gruppierung der Länder ist sehr schwierig zu interpretieren und die Zuteilung eines Landes in eine Gruppe ist zumindest nicht offensichtlich oder intuitiv. Auch die Betrachtung des Silhouette Plot zeigt mit einer mittleren Breite von 0.25, dass die gefundenen Strukturen nur sehr schwach ausgeprägt sind.

```
plot(silhouette(x=kmeans.countries$cluster,
                dist = dist(df.countries[,11:15])),
     cex.axis = 0.7, cex.lab = 0.7, cex.main = 0.6)
```

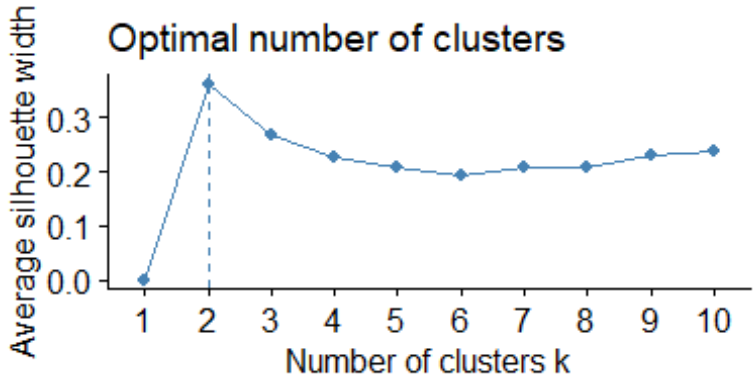


Zweiter Versuch: K-Medoids auf der Distanzmatrix *dist.countires*)

Als Basis kann dazu die Distanzmatrix aus der bereits ausgeführten Funktion *daisy* verwendet werden. Damit werden auch die Faktorvariablen zum Politischen System, der Region und Religion berücksichtigt.

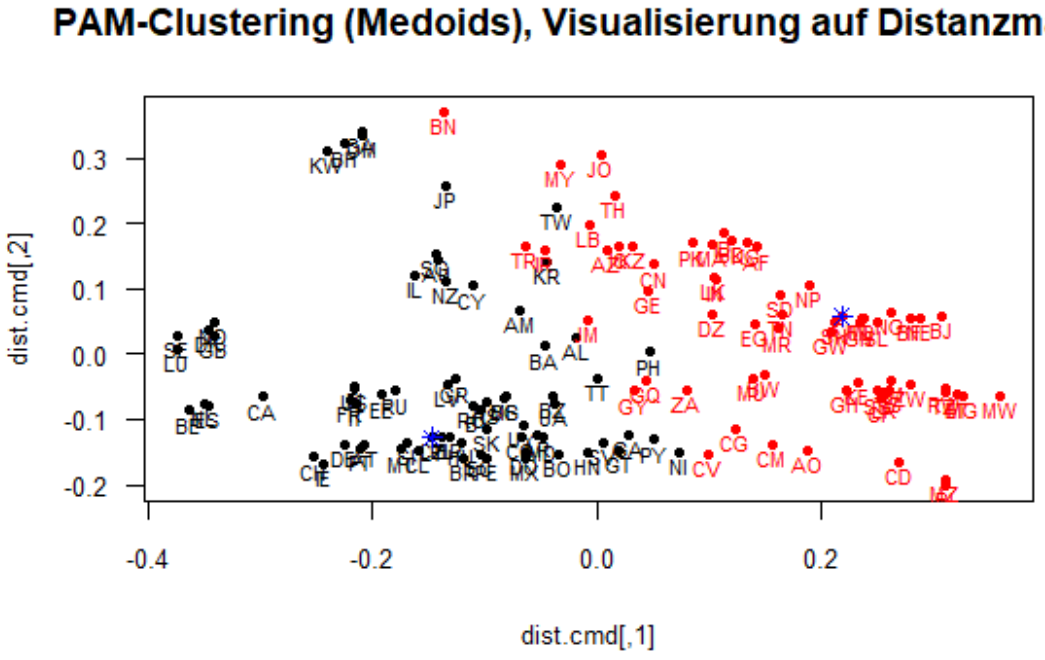
Auch hier muss als erstes die optimale Anzahl Cluster ermittelt werden, da dieser Parameter in der Clustering Funktion mitgegeben werden muss. Hier allerdings via der Silhouetten-Breite da die Bestimmung über WSS hier nicht geht).

```
fviz_nbclust(df.countries[ , 11:15],
             pam, method="silhouette",
             k.max = 10)
```



Der optimale Wert wird hier eindeutig mit 2 Clustern angegeben, womit immerhin eine durchschnittliche Silhouetten-Breite von etwa 4 erreicht werden kann.

```
# Clustering via pam
pam.countries <- pam(dist.countries, k = 2)
# Anzeige auf der Distanzmatrix (siehe Aufgabe 1, MDS)
plot(dist.cmd, col = pam.countries$cluster,
     las = 1, pch = 20, cex.axis = 0.8, cex.lab = 0.8,
     main = "PAM-Clustering (Medoids), Visualisierung auf Distanzmatrix")
text(dist.cmd[,1], dist.cmd[,2] - 0.02,
     labels = df.countries$ISOCode,
     cex=0.7, col = pam.countries$cluster)
points(dist.cmd[pam.countries$id.med,], pch=8, col="blue")  # Medoids ploten
```



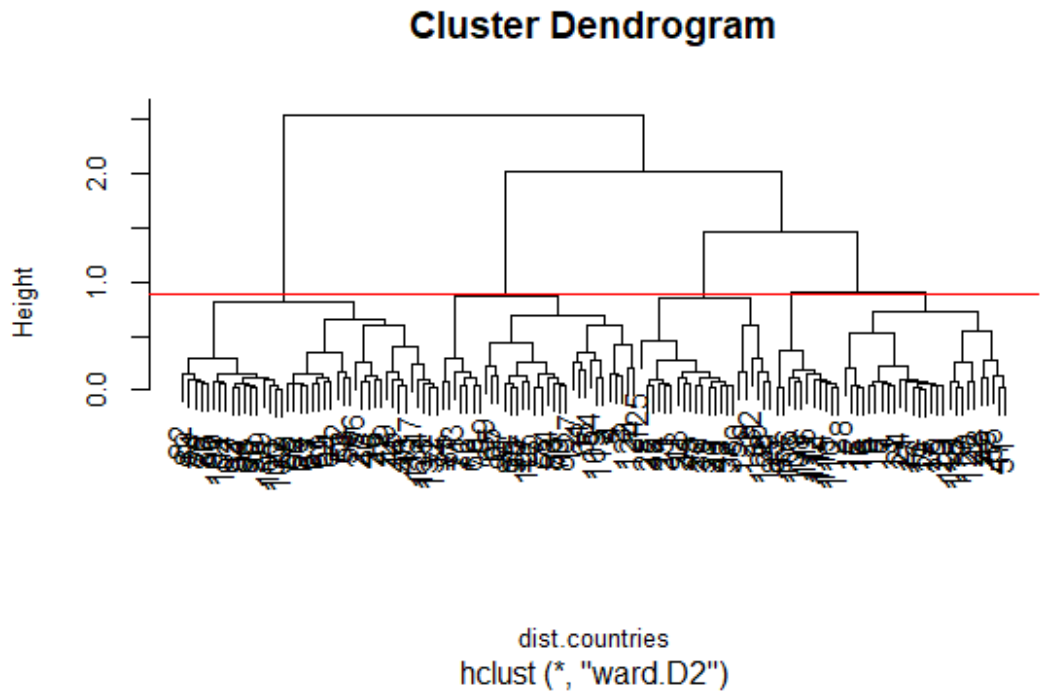
Damit können zwei gut voneinander getrennte Gruppen erstellt werden. Der zugehörige Silhouette Plot sieht auch gut aus mit dem bereits erwähnten Durchschnitt von 0.41, zeigt

aber auch ein paar negative Distanzen (plot wird hier nicht angedruckt). Eine inhaltliche Interpretation der Ähnlichkeit innerhalb der beiden Gruppen ist allerdings sehr schwierig, wenn nicht sogar unmöglich.

Dritter Versuch: hierarchiches Clustering auf Distanzmatrix

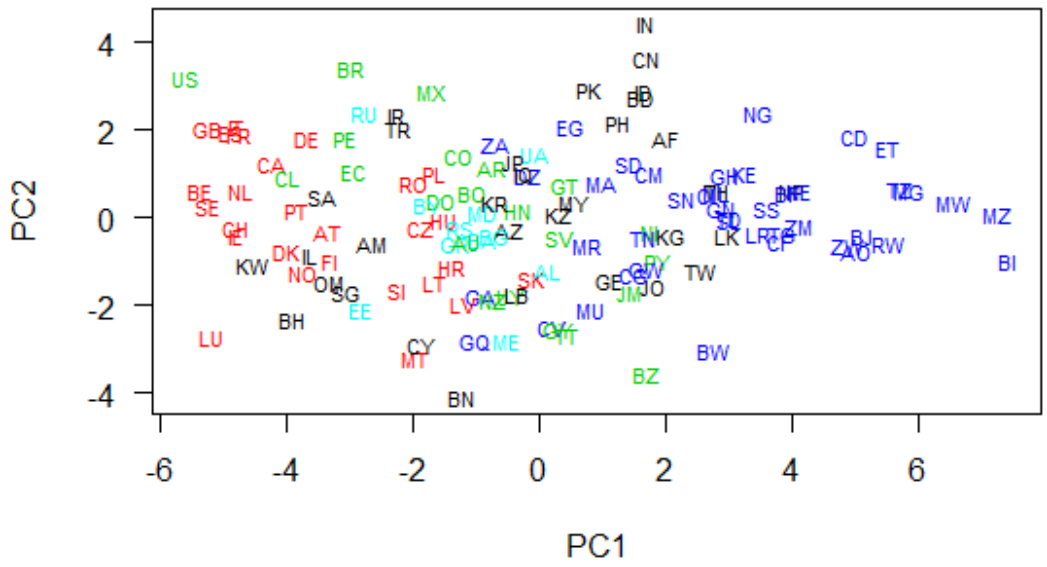
Als Alternative könnte das Clustering auch über eine hierarchisches Verfahren bestimmt werden. Bei der Erstellung des Baumes wird hier die Ward.D2-Methode für die Bestimmung der Distanzen zwischen den Gruppen angewandt.

```
hclust.countries <- hclust(dist.countries, method = "ward.D2")
plot(hclust.countries, cex.axis = 0.8, cex.lab = 0.8,)
abline(h=0.88, col="red")
```



Wenn man den Baum auf der Höhe 0.88 schneidet (viert-oberster Knoten: 0.8964933, fünft-oberster Knoten: 0.876533), erhält man 5 Cluster. Diese Anzahl wurde aufgrund der Resultate der bisherigen Analysen gewählt. Stellt man diese 5 Cluster via einer PCA dar, ergibt sich folgendes Bild.

```
grclust.countries <- cutree(hclust.countries, k = 5)
plot(pca$x, type="n",
     col = grclust.countries, las=1)
text(pca$x,
     labels = df.countries$ISOCode,
     cex=0.7, col = grclust.countries)
```

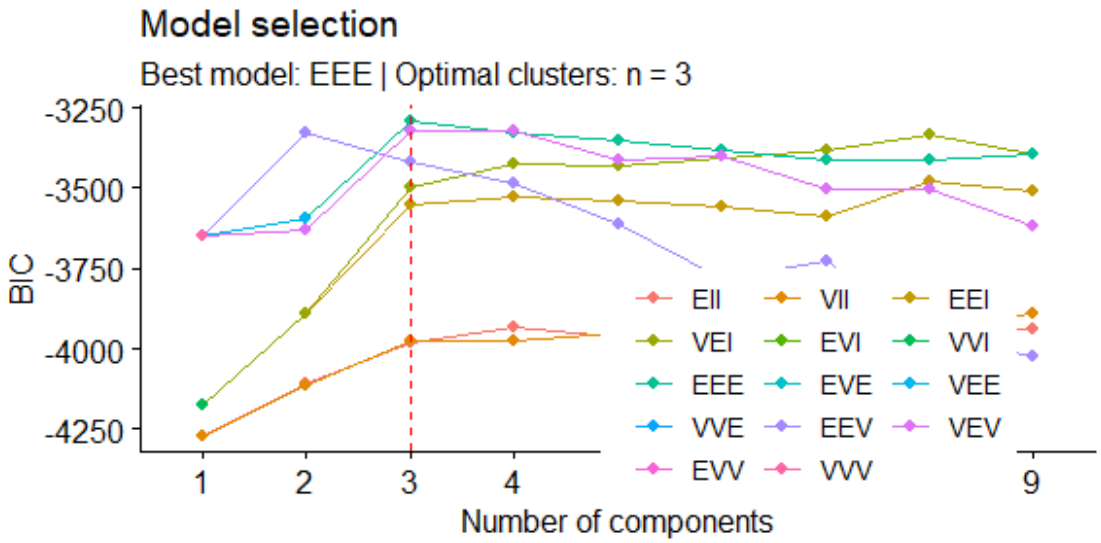


Dieses Clustering hat bislang am besten funktioniert und kann recht gut als eine Einteilung nach Region interpretiert werden. Bevor in der nächsten Aufgabe eine Klassifikation auf Region versucht wird, hier aber noch ein letzter, modellbasierter Clusteringversuch.

Letzter Versuch: Modellbasiertes Clustering auf *df.countries[,8:11]*

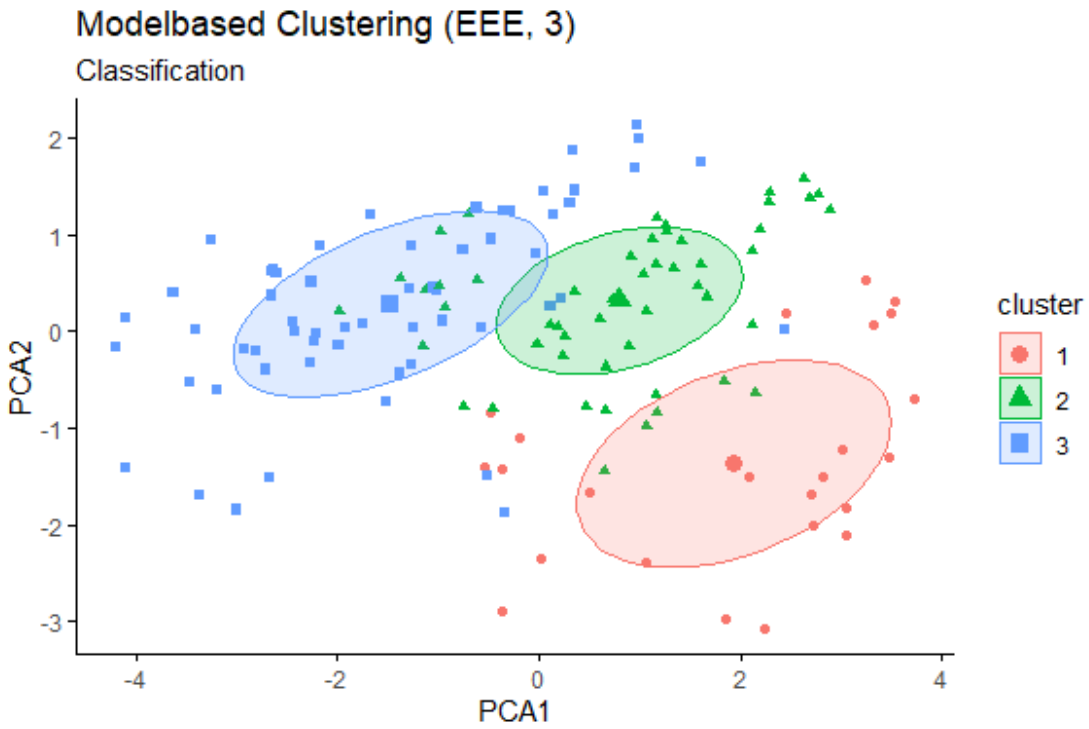
Zuerst wird über die Funktion *Mclust* das Optimale Modell und die optimale Anzahl Cluster bestimmt:

```
mclust.countries <- Mclust(df.countries[,8:15])
fviz_mclust_bic(mclust.countries)
```



Wenn man das vorgeschlagene Modell (EEE) mit 3 Clustern darstellt, ergibt sich allerdings wieder ein sehr schwierig zu interpretierendes Bild. Es ist mir hier zudem nicht gelungen die Ländercodes in der Grafik zu integrieren, was die Lesbarkeit praktisch verunmöglicht.

```
fviz_mclust(mclust.countries, "classification",
            geom= "point",
            main = "Modelbased Clustering (EEE, 3)",
            xlab= "PCA1", ylab="PCA2")
```



FAZIT zur Classifikation

Der einzige Anzatz der auch eine inhaltliche Interpretation der Daten zulässt ist das hierarchische Clustering, das zu einer recht guten Aufteilung der Daten nach Region führt. Allen anderen Clustering Methoden können zwar auf den Datensatz angewandt werden - viel Sinnvolles kommt dabei allerdings nicht heraus.

Klassifikation

Für die Klassifikation soll hier die Zielvariable *Region* anhand zweier Klassifizierer bestimmt werden. Als Klassifizierer werden hier zunächste die k-nearest neighbour Methode angewendet. Als zweite Methode kommt Random Forest zur Anwendung.

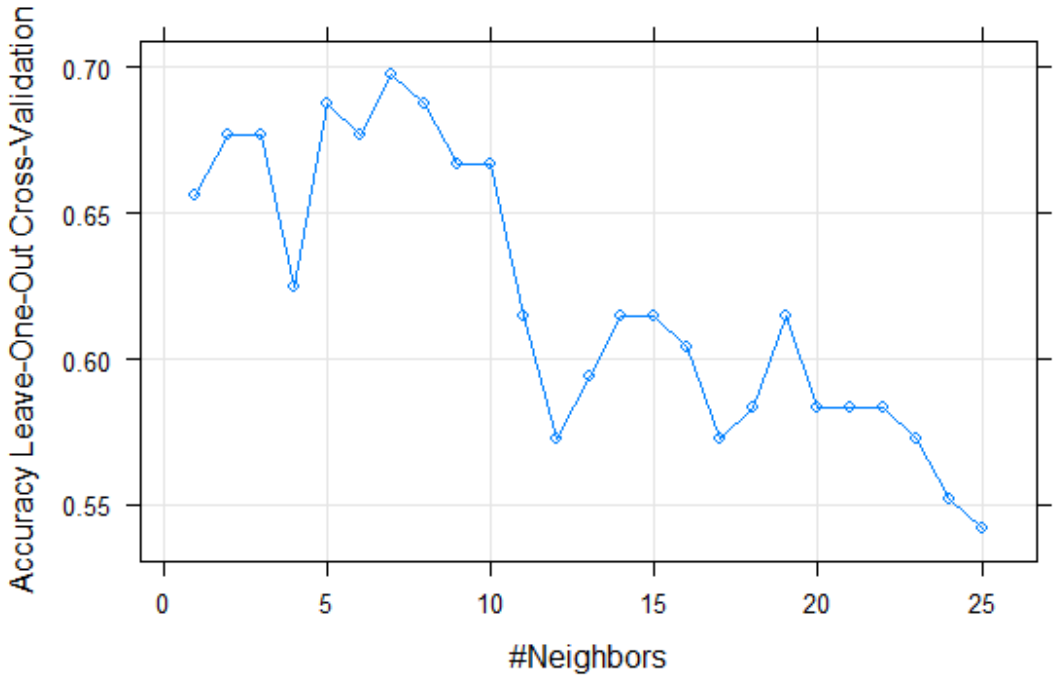
K-Nearest Neighbour

Zuerst wird der bisher verwendete Datendsatz um die Störvariablen bereinigt und als *df.countries2* weiterverwendet. Die numerischen Variablen sind somit weiterhin Log-transformiert. Dieser Datensatz wird dann in ein zufälliges Trainings- und Testdatenset (30%) aufgeteilt.

```
df.countries2 <- (df.countries[, 8:15])
# Generierung Training und Testset
set.seed(15)
rand <- createDataPartition(df.countries2$Region, p = 0.7, list = FALSE)
train.countries2 <- df.countries2[rand,]
test.countries2 <- df.countries2[-rand,]
```

Nun muss der Klassifizierer auf der Basis der Trainingsset an die Daten angepasst werden. Aufgrund des relativ keinen Datensets wird dabei die “Leave one out”-Kreuzvalidierung verwendet.

```
fitControl_knn <- trainControl(method = "LOOCV")         # Leave one out CV
set.seed(5)
class_knn <- train(Region ~., data = train.countries2,
                   method = "knn",
                   trControl = fitControl_knn,
                   tuneGrid = data.frame(k=1:25))
plot(class_knn)
```



Man sieht im Plot, dass der optimale Wert für die Anzahl Nachbarn bei k = 7 liegt. Der Test der Vorhersagen dieses Klassifizierers an den Testdaten kann in der Konfusionsmatrix dargestellt werden.

```
# Konfusionsmatrix für KNN Vorhersage auf Testset
set.seed(5)
pred_knn <- predict(class_knn, newdata = test.countries2)
confusionMatrix(dat = pred_knn, reference = test.countries2$Region)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Africa Americas Asia Europe Oceania
##   Africa       11        4    3      0       0
##   Americas      0        1    0      0       0
##   Asia          1        0    3      1       0
##   Europe        0        1    3     10       0
##   Oceania       0        0    0      0       0
##
## Overall Statistics
##
##                Accuracy : 0.6579
##                  95% CI : (0.4865, 0.8037)
##     No Information Rate : 0.3158
##     P-Value [Acc > NIR] : 1.535e-05
##
##                   Kappa : 0.5171
##
```

```
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Africa Class: Americas Class: Asia Class: Europe
## Sensitivity                 0.9167         0.16667     0.33333        0.9091
## Specificity                 0.7308         1.00000     0.93103        0.8519
## Pos Pred Value              0.6111         1.00000     0.60000        0.7143
## Neg Pred Value              0.9500         0.86486     0.81818        0.9583
## Prevalence                  0.3158         0.15789     0.23684        0.2895
## Detection Rate              0.2895         0.02632     0.07895        0.2632
## Detection Prevalence        0.4737         0.02632     0.13158        0.3684
## Balanced Accuracy           0.8237         0.58333     0.63218        0.8805
##                      Class: Oceania
## Sensitivity                      NA
## Specificity                       1
## Pos Pred Value                   NA
## Neg Pred Value                   NA
## Prevalence                        0
## Detection Rate                    0
## Detection Prevalence              0
## Balanced Accuracy                NA
```

Mit diesem Klassifizierer kann man also in 65.7894737% der Fälle (Accuracy) die Region eines Landes auf der Basis der restlichen Variablen korrekt ableiten. Bei der Beurteilung muss Ozenanien hier ausgeschlossen werden, da der Testsatz keinen diesbezüglichen Wert enthält. Am besten hätte man wohl ganz zu Beginn Ozeanien in der Klasse Asien integriert.

Die Sensitivität ist für Afrika und für Europa recht hoch, für Asia und für Americas aber gar nicht gut. Americas und Asien ist hingegen die Spezifität recht hoch mit Werten zwischen 93 und 100.

Random Forest

Für den Random Forest Klassifizierer können die untransformierten Originaldaten verwendet werden. Daher wird das Datensubset *df.countries3* erstellt und darauf das Trainins- und Testset definiert.

```
ohne.na2 <- complete.cases(countries[,c("Pop2016T", "GDPnomPC",
                                        "MilitSpendPC2016", "Covid1MPop",
                                        "CovidDead1Mpop", "CovidTest1Mpop",
                                        "IndepState", "PolitSystem", "Headofst
ate",
                                        "Region", "MainReligion")])
df.countries3 <- as.data.frame(countries[ohne.na2, c("Pop2016T", "GDPnomPC",
                                "MilitSpendPC2016", "Covid1MPop",
                                "CovidDead1Mpop", "CovidTest1Mpop",
                                "IndepState", "PolitSystem", "Headofstate",
                                "Region", "MainReligion")])
# Trainings- und Testdatenset Aufteilung
set.seed(1)
rand <- createDataPartition(df.countries3$Region, p = 0.7, list = FALSE)
```

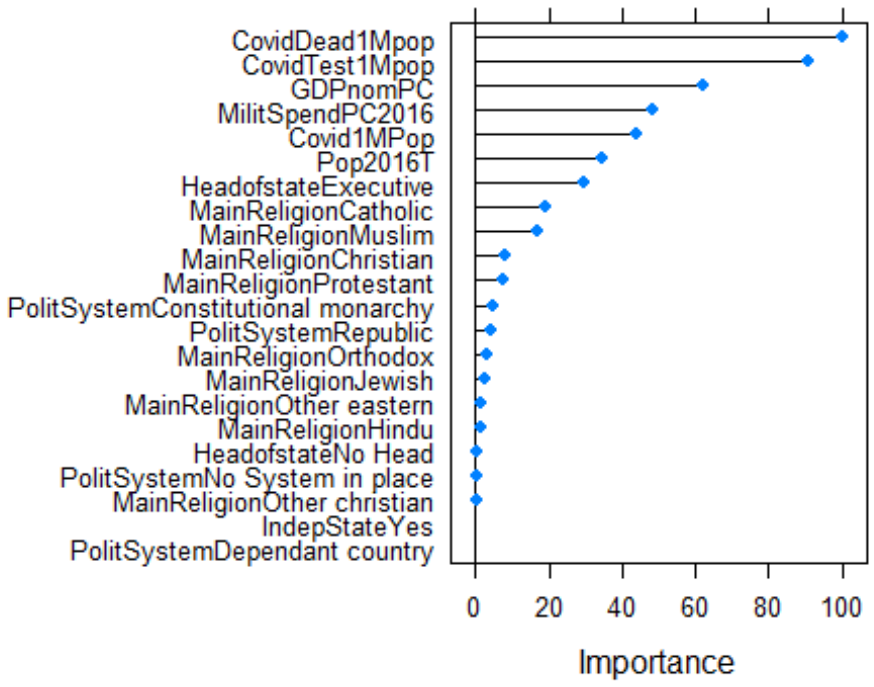
```
train.countries3 <- df.countries3[rand,]
test.countries3 <- df.countries3[-rand,]
```

Auf dem Trainingsset wird nun der Random Forest Klassifizierer mit der out of bag Methode angepasst.

```
fitControl_rf <- trainControl(method = "oob")     # out of bag
set.seed(5)
class_rf <- train(Region ~., data=train.countries3,
                  method="rf",
                  trControl = fitControl_rf,
                  tuneLength=3,                   # Anzahl getesteter Tuning P
arameter
                  ntree=1000)
```

Unter den massgebenden Variablen (*varImp(class_rf)*) für die Klassifikation fallen vor allem die verschiedenen Covid-Variablen auf, die insgesamt einen (zu?) starken Einfluss auf das Resultat haben.

```
plot(varImp(class_rf))
```



Das Resultat der Vorhersage des Klassifizierers wird auch hier wieder am Testset geprüft indem eine Konfusionsmatrix ersellt wird.

```
set.seed(15)
pred_rf <- predict(class_rf, newdata = test.countries3)
confusionMatrix(dat = pred_rf, reference = test.countries3$Region)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Africa Americas Asia Europe Oceania
##   Africa        9        1    1      0       0
##   Americas      0        4    0      0       0
##   Asia          3        2    9      3       1
##   Europe        1        0    0      8       0
##   Oceania       0        0    0      0       0
##
## Overall Statistics
##
##                Accuracy : 0.7143
##                  95% CI : (0.5542, 0.8428)
##     No Information Rate : 0.3095
##     P-Value [Acc > NIR] : 8.223e-08
##
##                   Kappa : 0.6164
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Africa Class: Americas Class: Asia Class: Europe
## Sensitivity                 0.6923         0.57143      0.9000        0.7273
## Specificity                 0.9310         1.00000      0.7188        0.9677
## Pos Pred Value              0.8182         1.00000      0.5000        0.8889
## Neg Pred Value              0.8710         0.92105      0.9583        0.9091
## Prevalence                  0.3095         0.16667      0.2381        0.2619
## Detection Rate              0.2143         0.09524      0.2143        0.1905
## Detection Prevalence        0.2619         0.09524      0.4286        0.2143
## Balanced Accuracy           0.8117         0.78571      0.8094        0.8475
##                      Class: Oceania
## Sensitivity                 0.00000
## Specificity                 1.00000
## Pos Pred Value                  NaN
## Neg Pred Value              0.97619
## Prevalence                  0.02381
## Detection Rate              0.00000
## Detection Prevalence        0.00000
## Balanced Accuracy           0.50000
```

Wenn die Zielvariable *Region* anhand dieses RF-Klassifikators vorausgesagt wird, kann auf dem Testset eine Accuracy von 71.4285714% erreicht werden.

Die Sensitivität ist hier für die Klassen Americas und Asien besser als bei KNN, uns bei Europa und Africa ist der Wert immer noch besser als 50%. Den tiefsten Wert sehen wir auch bei diesem Klassifizierer bei Asien. Die Spezifität über diese RF-Klassifiaktion liegt in einem ähnlichen Bereich wie bei KNN ist aber anders Verteilt: für Afrika und Europa sind die Werte besser, für Asien dagegen schlechter.

Insgesamt scheint das RF Modell aber insbesondere aufgrund der höheren Accuracy die bessere Wahl zu sein.