Prof. Dr. Steffen Wagner
Angewandte Statistik
Fachbereich II
Berliner Hochschule für Technik

Exercise 02: Higher Dimensional K-Means, Principal Components Analysi and K-Medoids

Machine Learning I – SoSe 24

1 Preparations

1.1 RStudio Project

1. Open your Machine Learning 1 RStudio Project
2. Create an R Script file to perform this exercise

1.2 Required Packages

For this exercise you require the following additonal R packages. Please make sure that you have installed them on your computer before coming to the workshop session since Eduroam is still not working.

```r
# check if packages can be loaded, i.e. they are already installed
library(rgl)       # 3D visualization
library(cluster)   # additonal cluster analysis techniques
library(ISLR2)     # material from book "Introduction to Statistical Learn."
```

If you get an error at this stage, you need to install the packages using

```r
install.packages("rgl")
install.packages("cluster")
install.packages("ISLR2")
```

2 High(er)-dimensional K-Means in R

2.1 K-means clustering with 3 dimensions

Copy and paste the following code into your R-script file.

```r
set.seed(135792468)
x <- matrix(rnorm(75*3), ncol = 3)
x[1:25, 1] <- x[1:25, 1] + 5
x[51:75, 2] <- x[51:75, 2] - 6
truth <- rep(1:3, each = 25)
pairs(x, col = truth)
```

```r
plot3d(x, col = truth, size=1, type="s")
```

a) Make sure you understand the above code.

b) `plot3d` opens a graphics window. You can rotate this 3D plot in different directions using the mouse to get different views. It help to make the window larger, to see the detail better. Try to find a viewing angle that separates the clusters well.

c) Run the `kmeans` function on this data with 3 clusters.

```r
km_out <- kmeans(x, centers = 3, nstart = 20)
```

What does the argument `nstart` do, and why is this used?

d) Compare the output clusters with the true cluster values with a 2-dim frequency table. How many observations are not clustered correctly?

```r
table(km_out$cluster, truth, dnn = c("cluster results", "true grouping"))
```

```
##                  true grouping
## cluster results  1  2  3
##                1  0  0 25
##                2  1 24  0
##                3 24  1  0
```

A: 2 observations are not clustered correctly. Note that the assignment of the numbers for the distinct clusters by kmeans is random.

e) Colour the clusters and add the cluster centroids.

```r
plot3d(x, col = km_out$cluster, size = 1, type = "s")
plot3d(km_out$centers, add=TRUE, col = 1:3, type = "s")
```

f) Repeat the above using just two clusters. How can you describe the two clusters in one sentence? One cluster consists of two of the original clusters.

g) Copy and adapt your code to do the following:

i) Generate a new matrix called `y` with 10 columns instead of 3, define the clusters in exactly the same way as above.

ii) Run `kmeans` with 3 centres on `x` and on `y`.

iii) Use the R function `table` to compare how many rows have been correctly and incorrectly assigned.

```r
set.seed(135792468)
y <- matrix(rnorm(75 * 10), ncol = 10)
y[1:25, 1] <- y[1:25, 1] + 5
y[51:75, 2] <- y[51:75, 2] - 6
truth <- rep(1:3, each = 25)
table(km_out$cluster, truth, dnn = c("cluster results", "true grouping"))
```

iv) Discuss the result.

Discussion: Since the additional 7 columns do not contain any grouping structure/information, since they are purely random, the cluster result is due to the structure in the already existing 3 columns and therefore does not change. Very little changes might be due to the overall random procedure used for data gegeration.

2.2 USA arrests data: PCA and clustering

James et al. considers the `USArrests` data frame, see the R help page `help(USArrests)`. Note that this is now an old dataset collected 50 ago.

The first part of this exercise closely follows the Lab *12.5.1 Principle Components Analysis* on page 530 of James et al. Afterwards you will use K-Means clustering on the original data and on the PCA data. The aims of this exercise are:

- to learn about the principal components method,
- to see how many clusters are appropriate,
- to investigate the characteristics of each cluster (if possible)
- and to see if it makes a difference between clustering the original data or clustering the principal components.

Note that there is no right answer here! Our main aim is to learn more about the data as opposed to proposing a good model.

Work through the following using *12.5.1* in James et al. to help you.

2.2.1 A few descriptive stats

a) Note the names of the four variables in `USArrests`.

b) For each variable obtain the mean and the standard deviation.

2.2.2 PCA

c) Use `prcomp` to get the principal components for this dataset.

```
data("USArrests")
pca_out <- prcomp(USArrests, scale = TRUE)
```

d) What is the difference between `pca_out$scale` and `pca_out$sdev`?

'scale' is the standard deviation of the inputed variables. 'sdev' is the standard deviation of the data projected onto the 1st, 2nd etd. principle comonent, i.e.

```
sd(pca_out$x[,1])
```

e) Create the *scree plot* and *cumulative variance plot* plots as in James. The first two Principal components explain 86.8% of the variance, so we will use just the first two.

```
explVar <- pca_out$sdev^2

plot(explVar, main = "Scree Plot",
     xlab = "PC", ylab = "Proportion of variance explained",
     ylim = c(0, max(explVar)), type = "b")
abline(h = 1, col = "red")

plot(cumsum(explVar)/sum(explVar), main = "Cumulative Variance Plot",
     xlab = "PC", ylab = "Cumulative Proportion of variance explained",
     ylim = c(0,1), type = "b")
abline(h = 0.8, col = "red")
```

f) Obtain the `biplot` using

```r
biplot(pca_out, xlabs = state.abb)
```

`state.abb` contains the standard US state abbreviations, giving a neater presentation, than that given in James et al.

2.2.3 K-Means with PCs

g) Run K-means on the PCA data with 2 clusters and plot the results. Note that the function `biplot` does not allow the points to be coloured according to cluster assignment. Make sure you understand the two plotting commands below.

```r
km_out <- kmeans(pca_out$x, centers = 2, nstart = 20)
plot(pca_out$x[, 1:2], type="n")
text(pca_out$x[,1], pca_out$x[,2], labels=state.abb, col=km_out$cluster)
```

How many states are in each cluster?

```r
table(km_out$cluster)
```

h) Adapt the code below to obtain the within sum of squares using 1 to 10 clusters plotting the results. At which value of k is there an "Elbow"?

```r
wss_vec <- rep(NA,??)
for(k in 1:??){
  km_out <- ??(??, centers=?, nstart=20)
  wss_vec[?] <- km_out$???
}
plot(???, type = "b")
```

```r
# correct code
wss_vec <- rep(NA, 10)
for(k in 1:10){
  km_out <- kmeans(pca_out$x, centers = k, nstart=20)
  wss_vec[k] <- km_out$tot.withinss
}
plot(wss_vec, main = "Scree Plot",
     xlab = "#clusters", ylab = "within SS", type = "b")
```

i) Using the "best" number plot the principal components coloured by cluster. As `USArrests` only has 4 variables obtaining a `pairs` plot is also worthwhile.

```r
km_out <- kmeans(pca_out$x, centers = 4, nstart=20)
pairs(pca_out$x, col = km_out$cluster)
```

j) Repeat parts g, h and i clustering on the original `USArrests` data frame, and compare the results.

```r
wss_vec <- rep(NA, 10)
for(k in 1:10){
  km_out <- kmeans(USArrests, centers = k, nstart=20)
  wss_vec[k] <- km_out$tot.withinss
}
plot(wss_vec, main = "Scree Plot",
     xlab = "#clusters", ylab = "within SS", type = "b")

km_out <- kmeans(USArrests, centers = 4, nstart=20)
pairs(USArrests, col = km_out$cluster)

table(kmeans(pca_out$x, centers = 4, nstart=20)$cluster,
      kmeans(USArrests, centers = 4, nstart=20)$cluster,
      dnn = c("PCA based", "Orig. data")) %>%
  addmargins()
```

2.3 Hamburg decathlon data, PCA and clustering

The decathlon data[1] contain the results of each discipline along with other information such as age, number of points gained for each competitor. The data file for the 2017 competition is provided in Moodle[2]. You should download and save this file in the usual way. The data is already pre-processed. In particular quite a few competitors did not complete all the events and so have missing values. Because of this the given data are restricted only to those who completed the entire decathlon. Note you will have to change the file-path in the `read.csv` command.

```r
# load data
decathlon <- read.csv("Data/DecathlonHamburg2017.csv")
```

a) Load the data.

b) The matrix scatter plot obtained using `pairs()` is not pretty, but we can nevertheless see some strong correlations. Which combination ofb) disciplines have a large positive or large negative correlation, and is this what you would expect?

   A: Running events are strong positive correlated with each other, throwing events are positive correlated with each other, jumping events are positive correlated with each other. Roughly: Running events are negatively correlated with throwing and jumping events. This makes sense as a small value is good in the running events.

c) Use K-Means directly on the unscaled data `decathlon` (use k= 4, 5 or 6). Plot the scatter plot matrix using colours for the clusters based on

```r
pairs(decathlon, col = ???$cluster)
```

```r
km_out <- kmeans(decathlon, centers = 4)
pairs(decathlon, col = km_out$cluster)
```

   Look at the plots in the 1500m column. The clusters are heavily dependent on the 1500m times. Why is this? This is an example where the data should be initially standardised using `scale(decathlon)`.

d) Use `prcomp` to get the principal components for this dataset and plot the first two using `biplot`. What characteristic is clear to see in the first principal component (x-axis)?

```r
pca_out <- prcomp(decathlon, scale = TRUE)
biplot(pca_out)
```

   Note also that there are 4 clear outliers.

   A: Running times are pointing in the opposite direction than the other disciplines.

e) Use

```r
plot3d(pca_out$x[,1:3], size = 1, type = "s")
```

   to create an interactive plot of the first three PCs. It is not so easy to learn anything more from this than the biplot tells us.

[1] https://www.10-kampf.de/ergebnisse/
[2] https://lms.bht-berlin.de/mod/folder/view.php?id=1188068&forceview=1

f) Create a *cumulative variance plot* for the principal components and decide about the number of PCs to use.

```r
plot(cumsum(pca_out$sdev^2)/length(pca_out$sdev),
     xlab = "PCs", ylab = "Cumulative Variance",
     main = "Cumulative Variance Plot",
     type = "b")
abline(h = 0.8, col = "red")
```

A: Using the first 3 PCs is probably a good choice.

g) Use k-means clustering on the principle components choosing an appropriate value for k and plotting the results in

i) a biplot and
ii) a scatter plot matrix.

```r
wss_vec <- rep(NA, 10)
for(k in 1:10){
  km_out <- kmeans(pca_out$x[, 1:3], centers = k, nstart=20)
  wss_vec[k] <- km_out$tot.withinss
}
plot(wss_vec, main = "Scree Plot",
     xlab = "#clusters", ylab = "within SS", type = "b")

# k=3 looks reasonable
km_final <- kmeans(pca_out$x[, 1:3], centers = 3, nstart=20)

# biplot coloured with cluster solution
plot(pca_out$x[, 1:2], col = km_final$cluster, main = "biplot")

# biplot coloured with cluster solution
pairs(pca_out$x[, 1:3], col = km_final$cluster)
```

Comment: In part (b) you ran K-Means on the original unscaled data, to emphasize the problems of the 1500m event. If you were to continue with a K-Means clustering on the original data, then scaling the variables is recommended.

3 K-Medoids using the PAM Algorithm

In this section you will need the packages `cluster` and `ISLR`. Using the package `cluster`, allows us to use some new types of clustering techniques and diagram to investigate the output from a clustering method. As `kmeans` was written before the `cluster` package, a `kmeans` object cannot be directly plotted by these new functions, although it is possible to *coerce* the `kmeans` object into a suitable form, see Exercise 3.2.

3.1 PAM Clustering

Use the following code in order to run the Partitioning Around Medoids PAM algorithm on the `USArrests` dataset from Exercise 2.2. You will need to replace `???` with the appropriate code.

The `USArrests` data set has the state names as row names. For the cluster plots it is better to use the state abbreviations. Replace the row names with the abbreviations.

```r
data("USArrests")
row.names(USArrests) <- state.abb
pairs(USArrests)
```

Have an initial look at the the PAM clustering with 4 clusters. Like before the data should be scaled before we start clustering

```r
library(???)
clustmat <- scale(USArrests)
pam_out <- pam(???, k = ???)
clusplot(pam_out, labels=3, col.p = pam_out$clustering)
```

```r
library(cluster)
clustmat <- scale(USArrests)
pam_out <- pam(clustmat, k = 4)
clusplot(pam_out, labels = 3, col.p = pam_out$clustering)
```

The cluster plot automatically plots the first two principal components. Each cluster is represented by an ellipse that encircles all the points in that cluster. The default setting for `pam`is to use the Euclidean distance metric. As we are considering robust clustering, compare the resulting clusters when using `manhattan`.

```r
pam_out <- pam(???, k = ???, metric = "manhattan")
```

```r
clusplot(pam_out, labels = 3, col.p = pam_out$clustering)
```

Use the Manhattan metric, when using `pam`for the rest of this workshop.

Another plot, used to assess the appropriateness of a clustering, is a **silhouette plot**. The elements are sorted into their clusters and its *silhouette width* is plotted. The silhouette width measures the similarity of each point to its cluster. A rough guide is a silhouette widths over 0.4 suggest the corresponding element is in a good cluster. Negative silhouette widths suggest that the corresponding element would probably be better assigned to a neighbour cluster. See the `silhouette` help page (or Wikipedia) for more details. The following code adds a reference line with the average silhouette width.

```
sp <- silhouette(pam_out)
plot(sp, col = 1:4)
mean(sp [, "sil_width"])
abline(v = mean(sp[, "sil_width"]))
```

Earlier you saw how the elbow method can be used as a guide assess to the best number of clusters. The mean silhouette width can also be used to do this.

```
avesw_vec <- rep(NA, 7)
for(k in 2:7){
  avesw_vec[???] <- mean(silhouette(pam(clustmat, k = ???, metric=???))[, "sil_width"])
}
plot(1:7, avesw_vec, type = "b", ylim = c(0,0.6))
```

```
avesw_vec <- rep(NA, 7)
for(k in 2:7){
  avesw_vec[k] <- mean(silhouette(pam(clustmat, k = k, metric = "manhattan"))[, "sil_width"])
}
plot(1:7, avesw_vec, type = "b", ylim = c(0,0.6))
```

The larger the silhouette width the better. Which number of clusters gives the largest mean silhouette width?

Re-run the PAM algorithm with the optimal number of clusters

```
pam_out <- pam(???, k = ???)
clusplot(???, labels = 3, col.p = pam_out$clustering)
sp <- silhouette(???)
plot(sp, col = 1:???)
abline(v = mean(sp[, "sil_width"]))
```

```
# completed code
pam_out <- pam(clustmat, k = 2)
clusplot(pam_out, labels = 3, col.p = pam_out$clustering)
sp <- silhouette(pam_out)
plot(sp, col = 1:2)
abline(v = mean(sp[, "sil_width"]))
```

Although none of the silhouette widths are large this silhouette plot is noticeably better than the first.

Compare the optimal PAM clustering with the K-Means result.

```
km_out<-kmeans(???, centers = ?? , nstart = 20)
table(km_out$???, pam_out$???)
```

```
km_out <- kmeans(clustmat, centers = 2, nstart=20)
table(km_out$cluster, pam_out$clustering)
```

The clustering obtained by the two methods is very similar.

The `pam`function tries to find a good starting point for the initial medoids. If you want to instead us multiple different random choices for the initial medoids, you can use the argument `nstart` with your chosen number of repeats. For this data set the two approaches give the same results.

3.2 Silhouette plot for K-means

The function silhouette was written for objects created by functions in the cluster package. kmeans is a function in the stats package installed and loaded with the base R installation. As a result
silhouette(km_out) doesn't work, but with a little work we can coerce the data into the correct format.

The first argument to silhouette should be the cluster vector accessible via km_out$cluster. The second argument should be a matrix containing the pairwise euclidean distance between each pair of points, dist(clustmat). E.g. the euclidean distance in the USArrests dataset between elements Alabama and Alaska is 37.18. (Note this is not geographical distance!).

The following code gives a silhouette plot for your last K-means clustering.

```r
# sp object by hand
sp <- silhouette(km_out$cluster, dist(USArrests))
plot(sp, col=1:???)
```

```r
# sp object by hand
sp <- silhouette(km_out$cluster, dist(USArrests))
plot(sp, col=1:2)
```