Exercise:
Multidimensional Data and Dimensionality Reduction

Introduction to Machine Learning with R

Exercise 1: Wisconsin breast cancer data

The Wisconsin data set describes features computed from digitized images of fine needle asprirates (FNA) of breast mass. In addition to the extracted features, the data set includes an ID number of the image and the diagnosis of the sample (1 = benign, 2 = malignant). For more information see https://www.kaggle.com/uciml/breast-cancer-wisconsin-data.

a) Load the "breastCancer_Wisconsin.rda" file using the `load('breastCancer_Wisconsin.rda')` command. The file contains the `wisco` data frame which stores the Wisconsin breast cancer data. First, take a look at the data. To do this, apply the `head` function to the data frame, which returns the first six rows of each column.

b) What are the dimensions of the data set, i.e. how many rows and columns are there? (**Hint**: `dim()`)

c) Next, apply the `str` function to the data frame. Try to understand what the output of the function means. Why could it be useful?

d) The main variable of interest, the `diagnosis` variable, is a categorical variable which tells us whether an image shows a benign or malignant tumor. However, as we can see in the output from the previous exercise, it is coded as a `numerical` variable (with the value 1 indicating a benign and the value 2 indicating a malignant tumor). This is usually not advisable, because it makes working with the categorical variable more difficult. For example, we always need a legend to understand the meaning of the numbers (1 = benign, 2 = malignant). In the worst case, using numbers for categorical variables can lead to a wrong analysis because a function might mistakenly treat a categorical variable as if it were continuous. The intended format in R for categorical variables is `factor`. Turn `diagnosis` into a `factor`, with the value 1 taking on the label `benign` and the value 2 taking on the label `malignant` (**Hint**: `wisco$diagnosis <- factor(...)`). Run again the `str` command to check if the conversion worked.

e) The picuture-ID (`id`) is coded as an `integer` (whole numbers). Although this will not affect the work in this exercise, it is again not advisable to code such a categorical ID-variable as numbers. Turn the `id` variable into a `factor`. Since we don't need to give new labels, you can leave the `labels` and `levels` arguments away when applying the `factor` function. (**Hint**: `wisco$id <- factor(...)`)

f) Perform a Principal Component Analysis (with scaling) on the data. Do not include the `id` factor in the PCA. Also exclude the `diagnosis` factor from the PCA. Why can't we include these factors in the PCA? Look at the created object. (**Hint**: `prcomp(..., scale.=TRUE)`)

g) Check the results of the PCA. What proportion of the variance is explained by the first PC? What proportion of the variance is explained by the first four PCs together? (**Hint**: `summary()`)

h) Perform a dimensionality reduction by looking only at the first two PCs. Plot the values of the two first PCs in a scatterplot.

i) Extend the plot by colouring the points according to the `diagnosis` variable. To do this, give the `diagnosis` vector as input to the `col` argument of the `plot` function (**Hint**: `plot(..., col=...)`). What can you see? We can also add a legend to show us the meaning of the colours. Check the solution and try to understand how the `legend` function works.

j) We now want to see how the 2D representation changes when no scaling is used. Perform the PCA again but without scaling and create the 2D plot of the first two PCs. What can you observe?

k) Using the PCA generated with scaling, we wish to find out how many PCs are needed to explain the data well. Create a scree-plot and interpret it.

l) We might be interested which variable contributes most strongly to the first PC. Plot the variables' loadings to the first PC (take absolute values of loadings) in a barplot. (**Hint**: `barplot()`, `abs()`, `pca_object$rotation`)

m) Which of the variables has the largest (absolute) loading on PC1? How large is this loading? (**Hint**: `which.max()`)

n) To make the situation a bit easier to oversee, we decrease the size of the data set by only working with the first eight columns. Create a new data set consisting only of those columns.

o) Try to get a visual impression of the data by plotting it in a pairs plot. You can use the `ggpairs` function (you need to load the `ggplot2` and `GGally` package for the function to work). The `id` variable has to be removed before applying the function. Interpret the created figure. (**Hint**: `library(ggplot2)`, `library(GGally)`, `ggpairs(...)`)

p) Perform a PCA with the reduced data (again excluding `id` and `diagnosis`, use scaling). Create a biplot of this PCA, showing the projections of the original variables. Which of the variables has the strongest contribution to PC2?