

Question 3.1:

To get a better understanding of PCA, let us first apply PCA on a low dimensional data set called `trees`, which is available in Base-R. The data set contains three variables: the height of a tree (in feet), the girth (or diameter in inches) and the volume (in cubic feet) of a tree. For the purpose of this question, we convert the continuous volume variable to a categorical variable (a factor). A tree is considered large if its volume is bigger than 25 cubic feet, and small otherwise.

- (a) First load the data set `trees` into R using the command `data(trees)` and create the categorical variable from the volume of tree using the command
`vol_fac <- as.factor(ifelse(trees$Volume > 25, "large", "small"))`. To visualise the data, use the R function `ggplot` to plot the height versus the girth for trees and size the dots according to their volume. Try to use a colour aesthetic to distinguish “large” and “small” trees.
- (b) Scale the height and girth variables using the R function `scale` to have mean 0 and variance 1, and then apply the PCA on the scaled height and girth variables using the R function `prcomp`. Interpret the PCA results.
- (c) Visualise the principal components (PCs) from Part (b) using the R function `ggplot`. For this, first scale the PC loadings by their standard deviations (singular values) to project them back to the original data space, and also transpose the rotation matrix so that the variables are in columns and PCs in rows to overlay them on the original `trees` plot.
- (d) Visualise the PCA results with `ggbiplot` (instead of `ggplot`) that is available in the R package `ggbiplot` which can be installed from github and loaded using the following commands:

```
library(devtools)  
install_github("vqv/ggbiplot")  
library(ggbiplot)
```

- (e) Now repeat the PCA without scaling the variables and by changing the unit of height from inches to millimeters. Does anything look wrong in the PCA result in this case when variables have very different scales?

Question 3.2:

We now apply the PCA on a high dimensional data set called `tumor_tissue`, which is available on Ultra. The dataset contains gene expression levels in 40 tumour and 22 normal colon tissue samples. It includes a total of 6500 human genes obtained using the Affymetrix oligonucleotide array. A main goal of the study was to find the best subset or combination of genes for detecting tumourous tissue. We here use 2000 genes with the highest minimal intensity across the samples. Note that the data set here contains a variable (the first column) named Y with the values t and n indicating whether the sample came from tumourous (t) or normal (n) tissue. The other 2000 columns are the 2000 genes.

- (a) First load the data set `tumor_tissue` into R environment using the command
`load(file="tumor_tissue.rda")`. Then, create a vector Y from the first column of data representing the response variable and a matrix X from all the other columns of data denoting the covariates or predictors. Scale X so that all its columns have mean 0 and variance 1.
- (b) Perform a singular value decomposition (SVD) on the scaled matrix X using the R function `svd`. How many of the singular values are non-zero? Is this in agreement with the theoretical results on SVD for high dimensional data in the lecture notes? Store the scores of the PCs in a matrix Z .
- (c) Now apply the PCA on the scaled data matrix X using the R function `prcomp`. Interpret the PCA results. How many principal components would explain about 90% of the data variance? Is this any useful here in terms of dimensionality reduction? Get a “scree plot” showing the percentage of variance captured by PCs, using the R function `fviz_eig` available in the R package `factoextra` which needs to be installed.
- (d) Use the R function `fviz_pca_var`, also available in the R package `factoextra`, to get a biplot visualising the similarities and dissimilarities between the samples and showing the impact of variables on the first and second PCs.
- (e) As an alternative way of getting the biplot in Part (d), similar to Question 3.1, use the R function `ggbio` to visualise the PCA results.
- (f) Plot histograms of the loadings of the first and second PCs and comment which loadings are the most important. For this, use the R function `hist` and try to specify an appropriate value for the `breaks` argument.

- (g) Note that this part of the question shall be tried later as it concerns “sparse PCA” to be covered in the next lecture. Considering the computation method for sparse PCA based on the elastic net as in the lecture notes, use the R functions `glmnet` and `cv.glmnet` to select an appropriate number of non-zero loadings for the first and second PCs. For this, set `alpha=0.5` in the elastic net models and use Z_1 and Z_2 as the response variables (i.e., fit two separate models) where Z_1 and Z_2 are the first and second columns of the scores matrix Z already stored in Part (b). To see how many features are important for each fit, look at the coefficient profile plots.