Statistical Data Mining II - Homework 2

Ezgi Karaesmen

March 10, 2016

Exercise 1: Carrying out a Principal Component Analysis (PCA) of Swiss bank note measurements

PCA is an exploratory data analysis method that helps us to summarize the variability and correspondence between the variables of the high dimensional data. This is achieved by reducing the dimensionality to a smaller number of representative variables, called principal components, that collectively capture and present most of the variability in the original data set. PCA is a very useful tool, as examining the data by visualizing two-dimensional scatterplots of all possible pairs of variables can be very time consuming and hard to interpret, and PCA overcomes this problem.

The data set used in this exercise consists of six variables and 200 observations, where first 100 observations were obtained from genuine bank notes and second 100 observations were obtained from counterfeit bank notes. Before applying PCA to the example data set, it would be helpful to look at the scale and variance of the variables, since the data set is not very high dimensional and can be helpful in deciding whether the data set should be scaled prior to PCA.

```
load("./SwissBankNotes.RData")

# Variance of each variable
apply(SwissBankNotes, 2, var)
```

```
##       length  height.left height.right  inner.lower  inner.upper
##    0.1417930    0.1303394    0.1632741    2.0868781    0.6447234
##     diagonal
##    1.3277163
```

```
# Boxplots to visualize the distribution and scale of the variables
par(mfrow=c(1,6), mar=c(2,3,2,1))
for(i in 1:6){
boxplot(SwissBankNotes[1:100,i], SwissBankNotes[101:200,i],
        names=c("Gen", "Count"),
        col=c("#5ab4ac" , "#d8b365"), main=colnames(SwissBankNotes)[i])
}
```

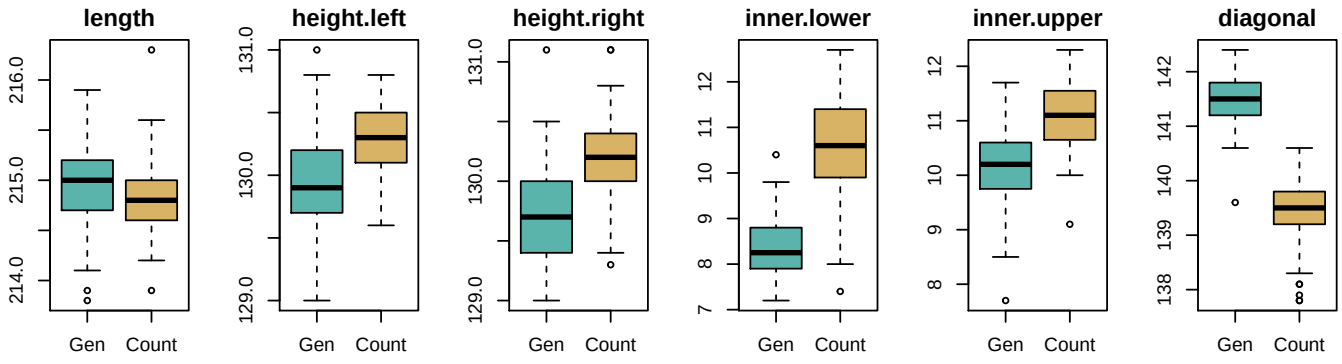

Figure 1: Box plots of each variable in the data set, visualized for genuine (Gen) and counterfeit (Count) observations.

As shown in Figure 1, the scales of the variables show high variability, especially for `inner.lower` and `inner.upper` variables compared to the rest. This indicates that the principal component analysis would require to shift the variable distributions, and to center to have mean zero. Furthermore, variance values of the variables seems less different yet show a variability, indicating that scaling each individual variable to have standard deviation one would be good practice to obtain better PCA results.

```
# Add factor column defining the type of the bank note
# and modify column names for a better biplot visualization
bnotes <- SwissBankNotes
bnotes[[7]] <- as.factor(c(rep("genuine", 100), rep("counterfeit", 100)))
colnames(bnotes) <- c("lenght", "height.L", "height.R", "inner.L", "inner.U", "diagonal","type")
head(bnotes)
```

```
##   lenght height.L height.R inner.L inner.U diagonal    type
## 1  214.8    131.0    131.1     9.0     9.7    141.0 genuine
## 2  214.6    129.7    129.7     8.1     9.5    141.7 genuine
## 3  214.8    129.7    129.7     8.7     9.6    142.2 genuine
## 4  214.8    129.7    129.6     7.5    10.4    142.0 genuine
## 5  215.0    129.6    129.7    10.4     7.7    141.8 genuine
## 6  215.7    130.8    130.5     9.0    10.1    141.4 genuine
```

```
# generating PCs for genuine observations,
# and compare centered, scaled PCs to non-centered or scaled
gen.pca.csF <- prcomp(bnotes[1:100,1:6], center = FALSE, scale = FALSE)
gen.pca.csT <- prcomp(bnotes[1:100,1:6], center = TRUE, scale = TRUE)

# Summary of genuine PCA without centering or scaling
summary(gen.pca.csF)
```

```
## Importance of components:
##                          PC1     PC2    PC3    PC4   PC5    PC6
## Standard deviation     318 0.82711 0.5588 0.3025 0.288 0.2057
## Proportion of Variance   1 0.00001 0.0000 0.0000 0.000 0.0000
## Cumulative Proportion    1 0.99999 1.0000 1.0000 1.000 1.0000
```

```
# Summary of genuine PCA wit centering and scaling
summary(gen.pca.csT)
```

```
## Importance of components:
##                            PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.4845 1.3026 0.9827 0.76348 0.57156 0.47340
## Proportion of Variance 0.3673 0.2828 0.1610 0.09715 0.05445 0.03735
## Cumulative Proportion  0.3673 0.6501 0.8111 0.90820 0.96265 1.00000
```

```
par(mfrow=c(1,2))
screeplot(gen.pca.csF,  type="lines", main="Genuine Only \n center = FALSE, scale = FALSE",
          cex.main=.8)
screeplot(gen.pca.csT, type="lines", main="Genuine Only \n center = TRUE, scale = TRUE",
           cex.main=.8)
```

Summary of the PCA results indicates the importance of each principal component, where the most important component is displayed as first. Proportion of variance for each component is displayed as the percentage of the variability captured by each component. Similarly cumulative proportion indicates the what percent of the variability is explained by a certain component and its precedents, which have higher importance. Although there are no set

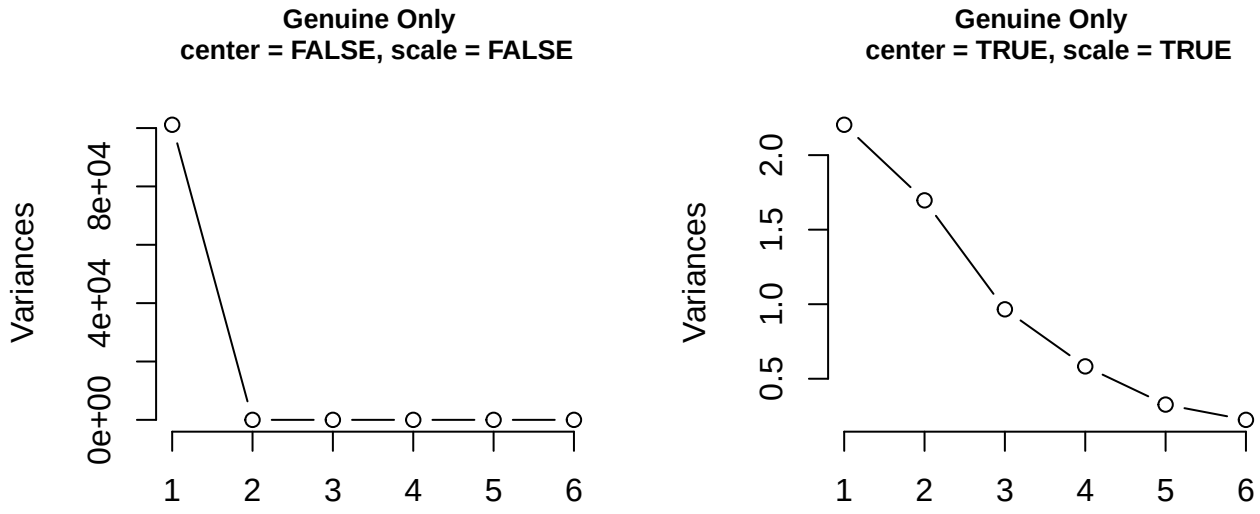

Figure 2: Scree plots showing the variance explained by principal components of only genuine observations. Left plot shows principal components generated without centering or scaling the data, whereas right plot shows upon centering and scaling

standards, first couple of principal components would be expected to explain most of the variability of the data, with first component having the highest proportion of variance and then decreasing gradually for the successive components. Visualization of the explained proportion by each component is achieved via scree plot in Figure 2. Usually an elbow effect is seen where the proportion of the variance reaches a low plateau after a certain component.

Summary results and scree plots for non-centered and centered genuine bank note data were produced and compared. Summary results of non-centered and centered data indicates that the centering is an important step of the PCA, since the non-centered data show an aberrant distribution of proportion of the variance, where all the variance is explained by the first component. This can also be seen in the scree plots where non-centered data has a very steep elbow. On the other hand, principle components of the centered and scaled data show an expected scree plot profile and distribution of proportion of the variance. First three components of the centered data captures about 80% of the variability, and the component 3 on the scree plot looks like a candidate for an elbow point, as successive components do not add much explained variability. Hence, first three component were selected for further analysis.

Centering and scaling was also preferred for the PCA of counterfeit and whole data sets. Summary of the PCA analysis and scree plots were produced.

```
count.pca.csT <- prcomp(bnotes[101:200,1:6], center = TRUE, scale = TRUE)
all.pca.csT <- prcomp(bnotes[,1:6], center = TRUE, scale = TRUE)

# Summary of PCA with counterfeit data
summary(count.pca.csT)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6
## Standard deviation     1.3915 1.3285 0.9941 0.8823 0.56755 0.45840
## Proportion of Variance 0.3227 0.2941 0.1647 0.1297 0.05368 0.03502
## Cumulative Proportion  0.3227 0.6169 0.7816 0.9113 0.96498 1.00000
```

```
# Summary of PCA with whole data
summary(all.pca.csT)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.7163 1.1305 0.9322 0.67065 0.51834 0.43460
```

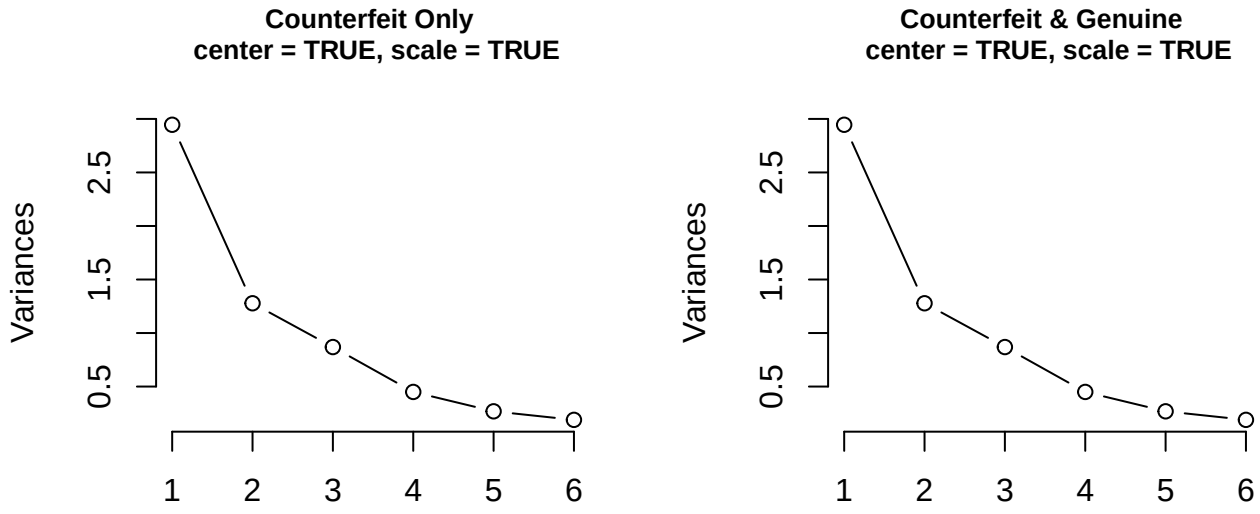

Figure 3: Scree plots of counterfeit and whole data sets

```
## Proportion of Variance 0.4909 0.2130 0.1448 0.07496 0.04478 0.03148
## Cumulative Proportion  0.4909 0.7039 0.8488 0.92374 0.96852 1.00000
```

```
par(mfrow=c(1,2))
screeplot(all.pca.csT, type="lines", main="Counterfeit Only \n center = TRUE, scale = TRUE", cex.main=.8)
screeplot(all.pca.csT, type="lines",
          main="Counterfeit & Genuine \n center = TRUE, scale = TRUE", cex.main=.8)
```

Similar to the results obtained from genuine data, summary and scree plots of PCA of both counterfeit and whole data sets, show that around 80% of variability is captured by the first three components, and the cut off according to scree plots would be the third or fourth component.

```
## the ggbiplot function used for biplot generation
## arguments : x (data), choices (components to be plotted),
##             grps (sub-groups in the data i.e. genuine, counterfeit),
##             leg.pos (legend position, or legen turn on/off)
##             main (plot title)

library(ggbiplot)
mybiplot <- function(x, comps, grps, leg.pos, main){
g<-ggbiplot(x, choices = comps, scale = 1, obs.scale = 1, var.scale = 1,
         groups = grps, ellipse = TRUE,
         circle = F, varname.size = 4,
         varname.adjust = 1.2, alpha=0.5)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal',
               legend.position = leg.pos, legend.text = element_text(size = 8),
               axis.title=element_text(size=10,face="bold"),
               plot.title = element_text(size = rel(1.5)))
g <- g + labs(title = main)
g

}

mybiplot(gen.pca.csT, comps=c(1,2), grps=F, leg.pos = "none", main="Genuine")
mybiplot(gen.pca.csT, comps=c(1,3), grps=F, leg.pos = "none", main="")
```

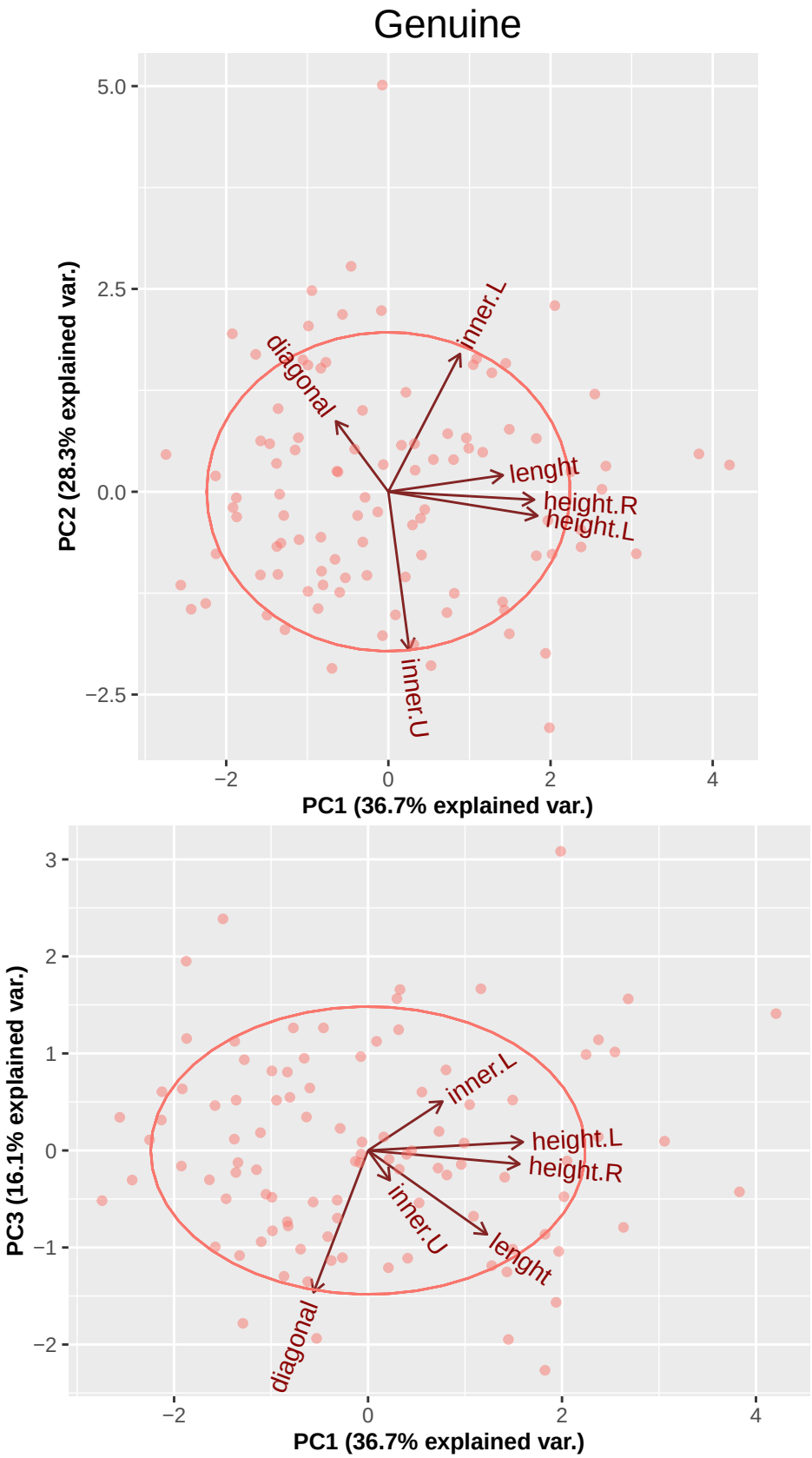

Figure 4: Biplots of the genuine data, produced with first three components. Components and percentage of the explained variable by the component is in indicated on the x and y axis.

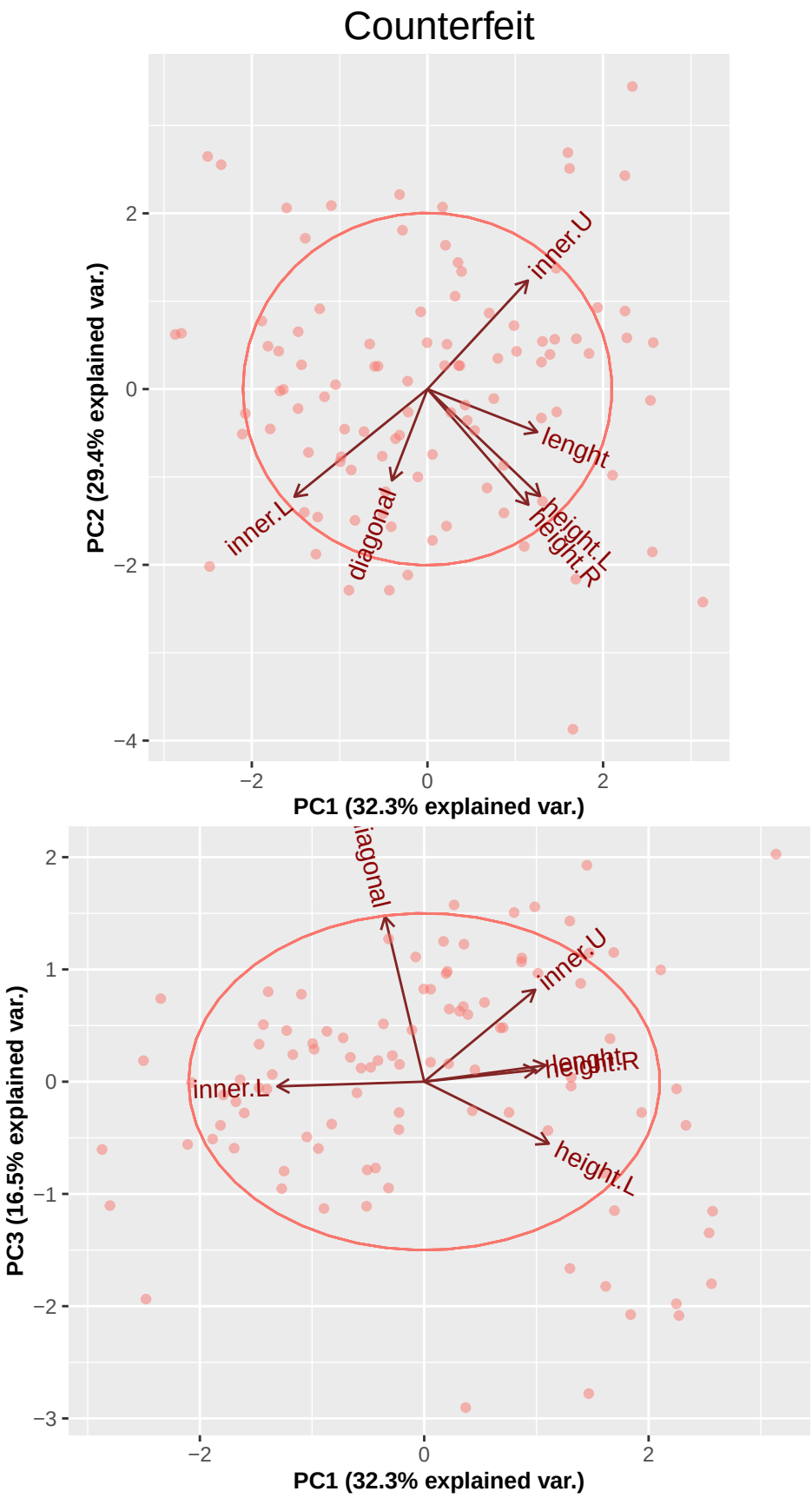

Figure 5: Biplots of the counterfeit data, produced with first three components. Components and percentage of the explained variable by the component is in indicated on the x and y axis.

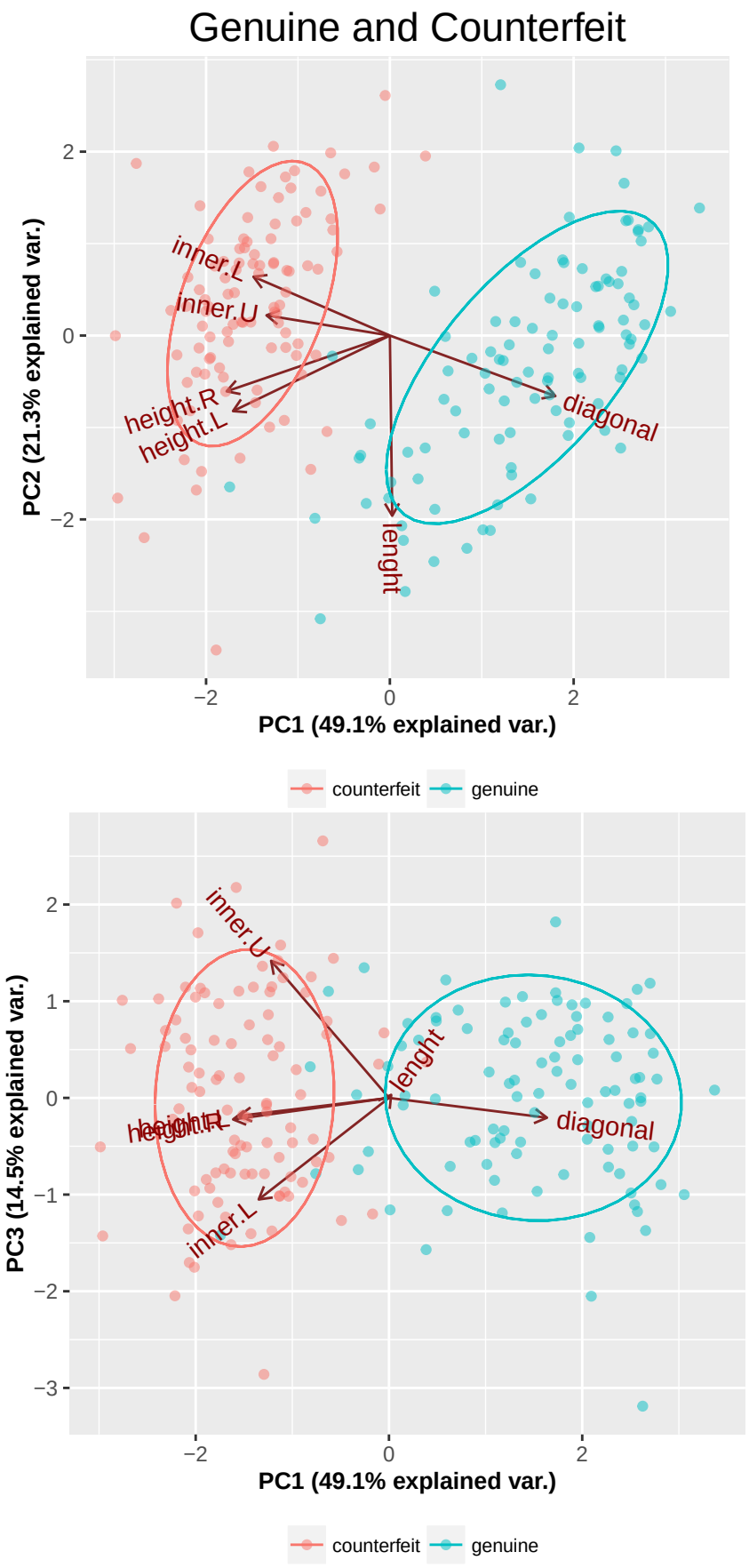

Figure 6: Biplots of the genuine and counterfeit data, produced with first three components. Components and percentage of the explained variable by the component is in indicated on the x and y axis.

Biplot is a two-dimensional scatter plot of principal components, where each point represents an observation on the data. Observations with similar variance profiles across variables will cluster closely, whereas different data subgroups with different variance profiles will be separated. As shown on biplots (Figures 4-6) data containing only genuine or only counterfeit bank note observations do not present a well defined separation, whereas the whole data set including both genuine and counterfeit observations show an more obvious separation, indicating that PCA successfully captures different variable profiles for genuine and counterfeit data.

Another element of the biplot represents loading of each variable on a component and is visualized with red arrows, indicating which component explains what proportion of variance of that certain variable. Additionally variables with smaller angles, that are closer to the same principal component and pointing out the same direction are likely to be correlated or have the similar variances. For example on the upper biplot in Figure 5, right height (`height.R`) and left height (`height.L`) are more parallel to the x-axis and located very closely, indicating these variables have a much higher loading for the first component and show a high positive correlation. This would make sense, since the bank notes have a rectangular shape, we would expect both right or left heights to be equal or, have a strong tendency to increase when one increases. Consequently, both of these variables were very closely located across all biplots, and were mostly represented by the first component. Another interesting variable, `diagonal` is mostly explained by the first component and seems to be an important contributor to the separation of the genuine and counterfeit sub-groups for whole data, whereas it shows distinct localisations and explained mostly by the second component for genuine and counterfeit only data, indicating that the diagonality is less variable within the two type of bank notes but show a significant variability between these two groups of bank notes, making it an important variable for the identification of the counterfeit bank notes.

Exercise 2: Generating simulated data, and performing PCA and K means clustering

(a) Generating simulated data set

Simulated data set consists of three different classes of observations, with each consisting of 20 observations, making it 60 observations in total. Each class and their observed values for 50 variables were generated with `rnorm()` function, with varying mean and standard deviations.

```
# (a) Generate a simulated dataset

cl.1 <- matrix(data=rnorm(50*20, mean = 3, sd = 1), nrow=20, ncol=50)   # Class 1
cl.2 <- matrix(data=rnorm(50*20, mean = 6, sd = 0.5), nrow=20, ncol=50) # Class 2
cl.3 <- matrix(data=rnorm(50*20, mean = 1, sd = 3), nrow=20, ncol=50)   # Class 3

sim.data <- rbind(cl.1, cl.2, cl.3)
# add a factor column indicating class
sim.data <- data.frame(sim.data, as.factor(c(rep("Class1", 20),
                                             rep("Class2", 20),
                                             rep("Class3", 20))) )

colnames(sim.data) <- c(paste("V", 1:50, sep=""), "Class")

sim.data[c(1:5), c(1:3, 49:51)] # First three rows and first three & last three columns
```

```
##         V1       V2       V3      V49      V50  Class
## 1 3.415783 1.479861 4.740629 4.406229 4.120138 Class1
## 2 2.482005 4.227340 1.490206 4.495931 2.902473 Class1
## 3 4.134846 3.342048 3.523320 4.746637 4.113102 Class1
## 4 3.804965 3.492961 3.048978 2.489217 1.459864 Class1
## 5 2.543909 3.731131 4.046873 1.358744 3.097912 Class1
```

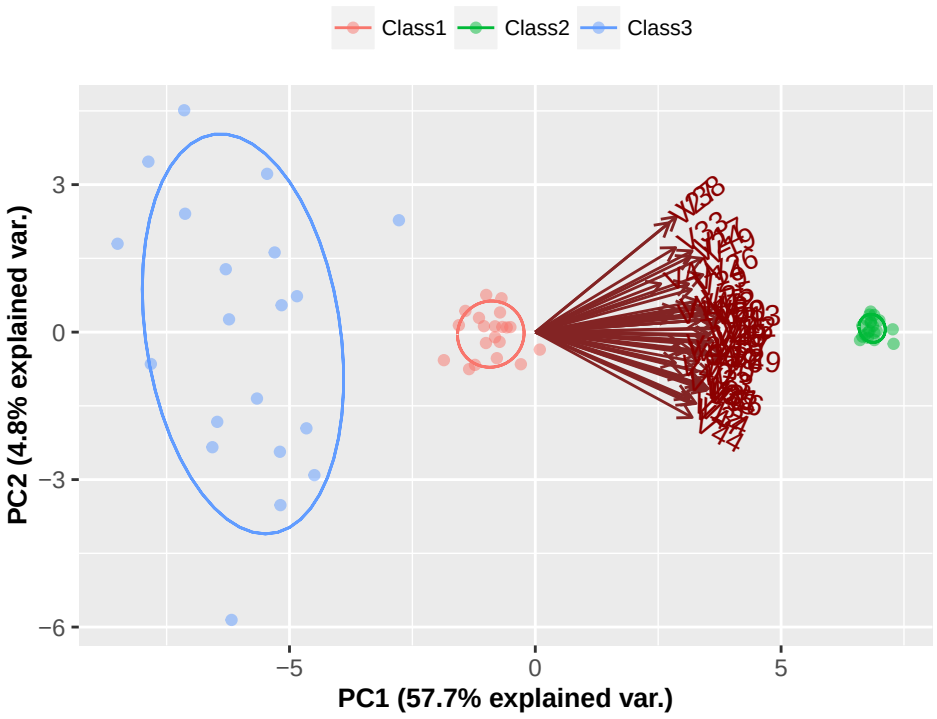

Figure 7: Biplot of simulated data

During the generation of the data, utilization of `set.seed()` function was forgotten, therefore data was saved as `simdata.RData` in order to achieve reproducibility and added to the homework dropbox.

(b) Perform PCA on the 60 observations and plot the first two principal component score vectors.

```
pca.sim.data <- prcomp(sim.data[,1:50], center = TRUE, scale = TRUE)
mybiplot(x=pca.sim.data, comps = c(1, 2), grps = sim.data$Class, leg.pos = "top", main = " ")
```

Different classes of data was nicely separated, representing the variability within each class. Class 1, 2 and 3 have means of 3, 6, 1, and standard deviations of 1, 0.5, 3 respectively. Therefore, as shown in Figure 7 class 3 has the most dispersed observations across all classes.

(c) Perform K -means clustering of the observations with K = 3

K-means clustering is a partitioning method that clusters the observations of a data set into desired number of K clusters. Each observation is assigned to only one of the clusters, making the clusters distinct and non-overlapping. This is achieved by minimizing the distance between each observation within a cluster. (1) Initially algorithm assigns a random cluster for each observation, (2) then computes a cluster center (centroid), a vector of the variable feature means for the observations for each cluster. (3) The distances between each observation and centroid are then calculated and the observations that are closest to the newly assigned cluster center are clustered in the same cluster. Steps 2-3 are repeated unlit reaching convergence, and there is no change of cluster centers. One of the problems of this algorithm is that while looking for a global optimum cluster center, algorithm can get stuck at a local optimum, and although the general k-means method has been empowered with other algorithms, preventing a quick convergence to a local minimum, trying several random starts is recommended.

```r
library("multtest")
library("fpc")
library("cluster")

# cluster for 3 groups,
# exclude factor column, indicating class
km3 <- kmeans(sim.data[,1:50], 3)

#tabulate the results
table(km3$cluster, sim.data$Class)
```

```
##
##      Class1 Class2 Class3
##   1       0     20      0
##   2       0      0     19
##   3      20      0      1
```

```r
# see the misclassified observation
tail( cbind(km3$cluster, as.character(sim.data$Class)) )
```

```
##    cluster  class
## 55       2 Class3
## 56       2 Class3
## 57       2 Class3
## 58       2 Class3
## 59       3 Class3
## 60       2 Class3
```

Upon K-means clustering for `K=3`, we can see that all the class 1 observations were clustered in cluster 3, all class 2 in cluster 1, almost all class 3 in cluster 2, with only one class 3 observation being misclassified in cluster 3 with all class 1 observations. This indicates that 98% of the simulated data was clustered as expected for each class. In order to visualize the localization of the clusters, cluster centers and observations, first two columns of the simulated data was plotted in a two-dimensional scatter plot, indicating the clustering of the observations with colors and actual class in numbers. This scatter plot should give us an idea about the clusters, since the variable values of each observation were randomly assigned from the same distribution for each class. As shown in Figure 8 (left), most of the observations that belong to the same class are clustered together accordingly, with a cluster center that visually seems correct. Only one class 3 observation is incorrectly clustered in cluster 3, a cluster that only class 1 observations are assigned. This is quite unexpected when we look at the plot as another class 3 observation localized very closely and clustered appropriately. However, this is just a representation of two variables, whereas the distances are computed by taking all variables into account. Therefore results of the k=3 clustering on the original simulated data set were also visualized with first principal components, generated in the part (a) of the exercise. Principal components 1 and 2 cumulatively explain about 60% (55% PC1 and 5% PC2) of the variance, which provides a much better a much better visualization (however one then fails to visualize data centers). As shown in Figure 8 (right), the misclassified data point is localized between the two obvious clusters that are relatively well separated across the x-axis, or PC1. It is likely that this observation is a candidate for an outlier and therefore has less distance to the center of cluster 3 (where all class 1 observations are clustered) then cluster 2 (where all class 3 observations are clustered). Same visualization methods were also applied to next part of the exercise.

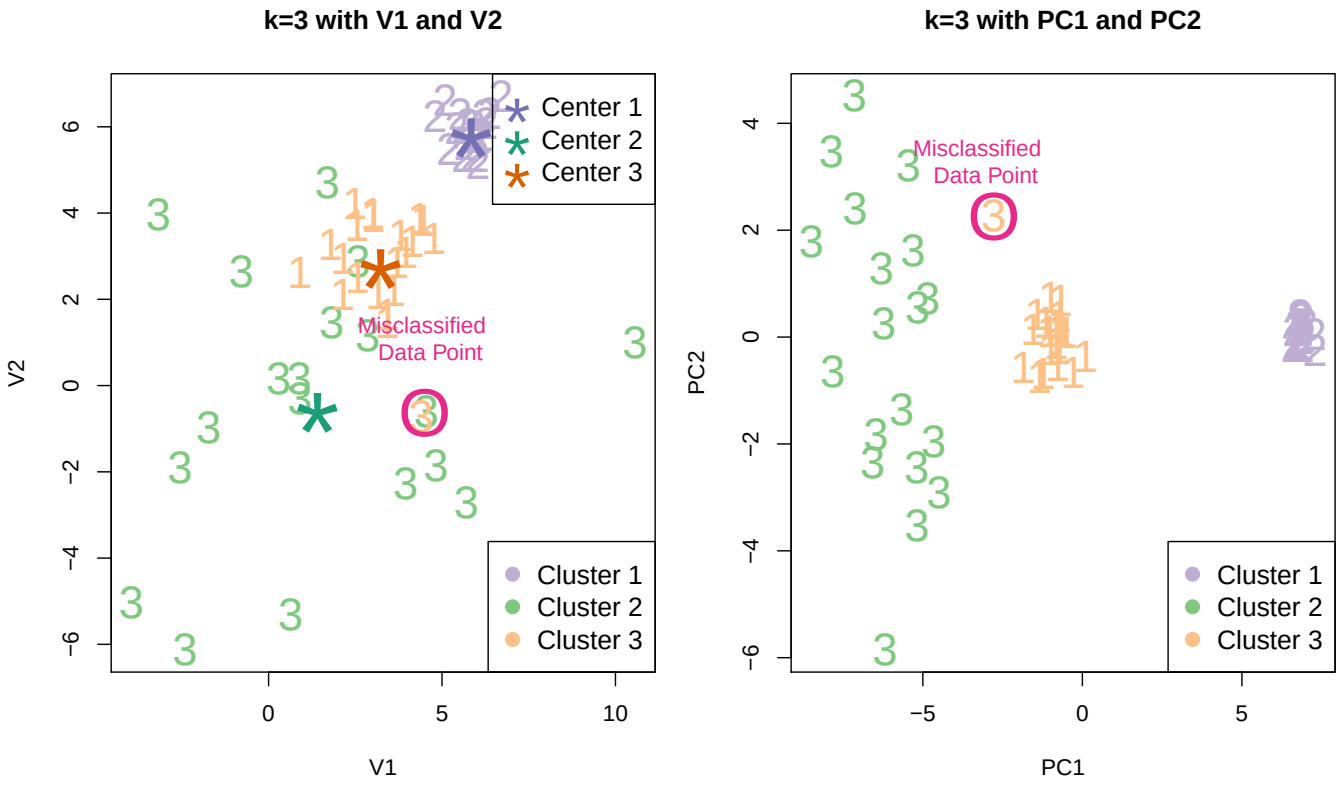

Figure 8: Scatterplot of simulated data variables V1 vs V2 (on the left) and principal components PC1 vs PC2 (on the right) of the simulated data. Each observation is represented by the actual class number they belong to. Each cluster assigned to the observation is represented with different colors as indicated in lower the legend. Each cluster center is represented with a star (*) and with a different color for each cluster as shown in the upper legend. In this example a class 3 observation is falsely clustered in cluster 3, a cluster that only class 1 observations are assigned. (R code for the figure can be found in appendix.)

(d) Perform K-means clustering with K = 2 and K = 4.

Same pipeline applied to k=3 clustering of the was also applied to k=2 and k=4

```
## (d) k =2 and k=4

# k = 2
km2 <- kmeans(sim.data[, -51], 2)
table(km2$cluster, sim.data$Class)
```

```
##
##      Class1 Class2 Class3
##   1     20      0     20
##   2      0     20      0
```

When K was set to 2, all class 1 and 3 observations were clustered in cluster 1 and all class 2 observations in cluster 2, indicating that class 1 and class 3 are closer compared to class 2. This is probably due to the set mean and variance of the classes. Since class 2 has the largest mean and the smallest standard variation, it separates very well from other classes, which have means of 3 and 1, and standard deviations of 1 and 3 for class 1 and 3 respectively. As shown in Figure 9, class 1 and 3 observations are localized much closer to each other and class 2 is well separated, resulting in clustering classes 1 and 3 together as expected when only clustered with 2 clusters.

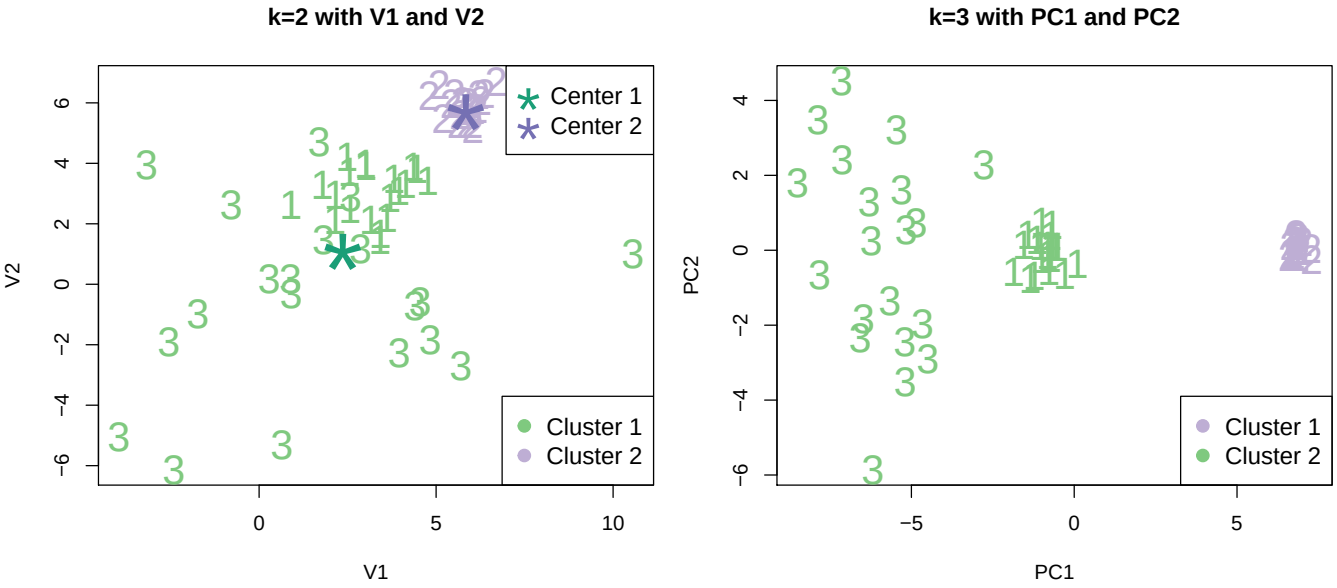

Figure 9: Scatterplot of simulated data variables V1 vs V2 (on the left) and principal components PC1 vs PC2 (on the right) of the simulated data. Each observation is represented by the actual class number they belong to. Each cluster assigned to the observation is represented with different colors as indicated in lower the legend. Each cluster center is represented with a star (*) and with a different color for each cluster as shown in the upper legend. (R code for the figure can be found in appendix.)

```
# k = 4
km4 <- kmeans(sim.data[, -51], 4)
table(km4$cluster, sim.data$Class)
```

```
##
##      Class1 Class2 Class3
##   1      0     20      0
##   2      0      0     16
```

```
##    3       0       0       3
##    4      20       0       1
```

When K is set to 4, all class 1 and class 2 observations are clustered under clusters 4 and 1 respectively. However observations of class 3, the class with the highest variance, are mostly clustered under cluster 2 and only a few are clustered under cluster 3 without sharing their clusters with other class observations. On the other hand, the same observation, namely observation 59, that was misclassified in part (c) was again clustered with class 1 observations under cluster 4. This again suggests that this observation is an outlier of class 3 and is much similar to class 1 observations. On the other hand the 3 class 3 observations that are assigned to cluster 2, seems like an artifact of setting an unnatural K (Figure 10).

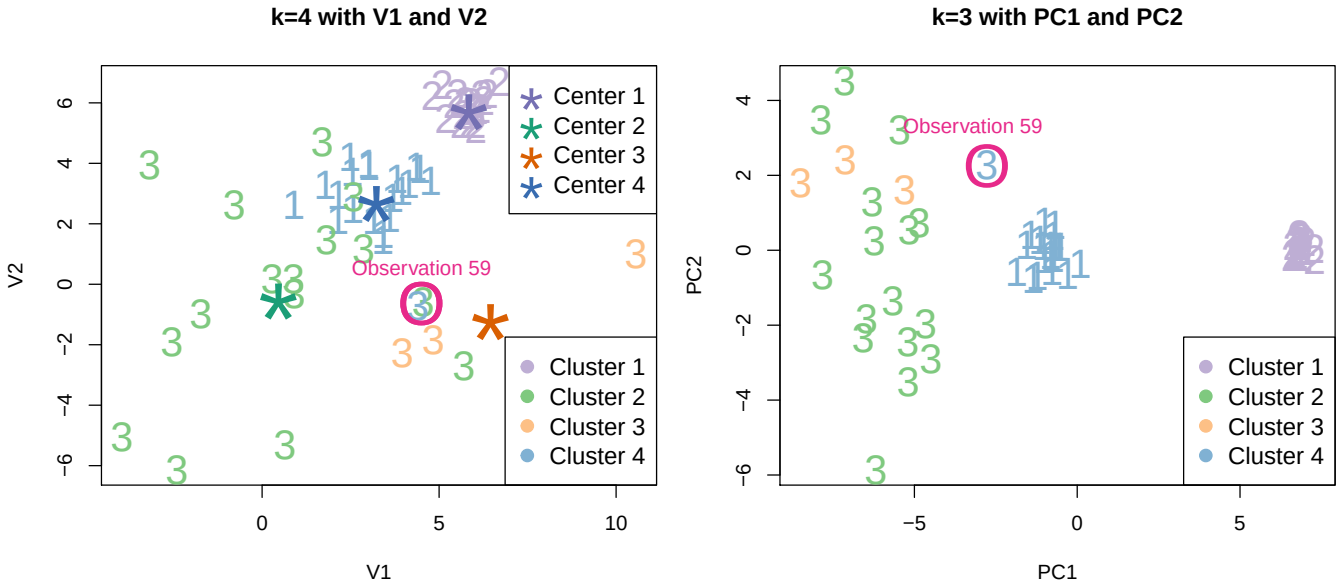

Figure 10: Scatterplot of simulated data variables V1 vs V2 (on the left) and principal components PC1 vs PC2 (on the right) of the simulated data. Each observation is represented by the actual class number they belong to. Each cluster assigned to the observation is represented with different colors as indicated in lower the legend. Each cluster center is represented with a star (*) and with a different color for each cluster as shown in the upper legend. (R code for the figure can be found in appendix.)

(e) Perform K-means clustering with K = 3 on the first two principal component score vectors

```r
## (e) Kmeans with comp1 & 2
# names(pca.sim.data)
pca.km3 <- kmeans(pca.sim.data$x[, 1:2], 3)
table(pca.km3$cluster, sim.data$Class)
```

```
##
##      Class1 Class2 Class3
##    1      0      0     19
##    2      0     20      0
##    3     20      0      1
```

```r
plot(pca.sim.data$x[,1:2], type="n", main="k=3 on PC1 and PC2")
text(pca.sim.data$x[cl2.1, 1:2], label=2, col='#beaed4', cex=2)
text(pca.sim.data$x[cl3.2, 1:2], label=3, col='#7fc97f', cex=2)
```

```
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], label=3, col='#fdc086', cex=2)
text(pca.sim.data$x[cl1.3, 1:2], label=1, col='#fdc086', cex=2)
legend("bottomright", legend = paste("Cluster", 1:3), col=c('#beaed4','#7fc97f','#fdc086'), pch=20, pt.cex
# see which one was mis-clustered
points(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], col = '#e7298a', pch = "O", cex = 3) #label=3, col='#
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2]+1.1, col='#e7298a',
     labels="Misclassified \n Data Point", adj=c(0.6, 0.6), cex=1)
```

Same exact clustering pattern that was observed raw data was observed using the first two principal component scores for each observation. This time although only two variables were included in the analysis, because of the special property of principal components, where the first components explain most of the variance in the data, we still get the same results. This indicates that principal components can be used for many other statistical methods other than biplotting, and are very useful for reducing the dimensionality of the data, slightly lifting the "curse of dimensionality".

(f) Using the `scale()` function, perform K -means clustering with K = 3 on the data after scaling each variable to have standard deviation one.

```
## (f) scale and cluster
scl.data <- scale(sim.data[,-51])
scl.km3 <- kmeans(scl.data, 3)
```

```
##
##     Class1 Class2 Class3
##   1     20      0      1
##   2      0      0     19
##   3      0     20      0
```

Also with scaling, same pattern was observed, and observation 59 was again incorrectly clustered, suggesting that reducing the variation within the this simulated data set does not improve clustering results. This also supports the fact that observation 59 cannot be normalized even upon scaling and keeps its distance from its actual class, class 3.

Exercise 3: Applying hierarchial clustering to `Ch10Ex11.csv` gene epression data set

(a) Loading the data

```
library("fpc")
library("cluster")

genedata <- read.csv("Ch10Ex11.csv", header=F)
colnames(genedata) <- c(paste("H", 1:20, sep=""), paste("D", 1:20, sep=""))
rownames(genedata) <- paste("Gene", 1:1000, sep="")

genedata[1:5, c(1:3, 38:40)]
```

```
##               H1         H2         H3        D18         D19        D20
## Gene1 -0.9619334  0.4418028 -0.9750051  0.4681471  1.06110000  1.6559700
## Gene2 -0.2925257 -1.1392670  0.1958370 -0.5423036  0.31293890 -1.2843770
```

```
## Gene3  0.2587882 -0.9728448  0.5884858  0.4016818 -0.01622713 -0.5265532
## Gene4 -1.1521320 -2.2131680 -0.8615249  0.0108553 -1.04368900  1.6252750
## Gene5  0.1957828  0.5933059  0.2829921  0.2007785 -0.67594210  2.2206110
```

(b) Apply hierarchical clustering to the samples using correlation-based distance, and plot the dendrogram.

Correlation based distance `as.dist((1 - cor(genedata))/2)` was applied to the gene expression values obtained from healthy and diseased individuals. Correlation-based distance clusters two observations together if their features are highly correlated, even though the observed values may be far apart in terms of Euclidean distance. This means that even the gene expressions are very different between clustered samples, the gene expression profile has a very similar behavior, where for example certain genes are distinctly up or down regulated for a certain disease state. Upon calculating the distance, hierarchical clustering was computed using `hclust()` function, using different linkage algorithms. Results were presented as dendrograms for every different linkage algorithm, which resulted in different clustering of the samples, suggesting that results depend on the type of linkage used. As shown in Figure 11 most of the algorithms successfully separated the healthy and diseased samples, with average linkage method providing a less accurate clustering. Linkage method determines how the dissimilarities between clusters or single observations are determined according to a given distance matrix. Therefore it is expected to have an effect on the actual clustering.

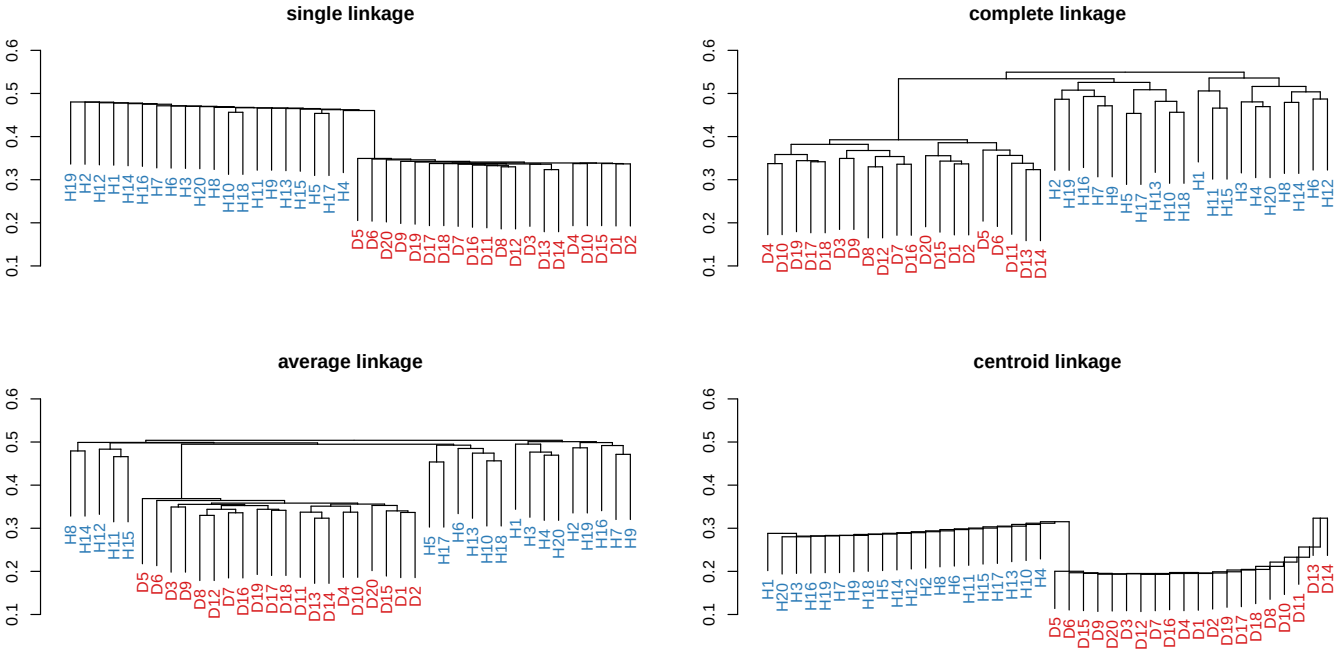

Figure 11: Hierarchial clustering of the gene expression data set for different linkage methods. Healthy and diseased samples were represented with blue and red colors respectively. Following correlation based distance 'as.dist((1 - cor(genedata))/2)' was preferred for the generation of the dendrogram, as suggested in '?dist' help file (R code for the figure can be found in appendix.)

(c) Determine which genes differ the most across the two groups

The genes with expression values that differ the most across groups can be determined by calculating the distance between diseased and healthy samples for a certain gene. However, to determine the significance of this distance, a multiple testing procedure should be applied, and the significantly different genes between healthy and diseased samples should be determined. Upon setting a false discovery threshold, and filtering insignificant genes, calculated distances can be sorted in descending order and top 100 genes with the largest distance can be determined as the most different genes between two groups.

```r
library("multtest")

class <- as.factor(c(rep("H", 20), rep("D", 20)))
mt.test <- mt.maxT(genedata, classlabel=class)
```

```
## b=100    b=200   b=300   b=400   b=500   b=600   b=700   b=800   b=900   b=1000
## b=1100   b=1200  b=1300  b=1400  b=1500  b=1600  b=1700  b=1800  b=1900  b=2000
## b=2100   b=2200  b=2300  b=2400  b=2500  b=2600  b=2700  b=2800  b=2900  b=3000
## b=3100   b=3200  b=3300  b=3400  b=3500  b=3600  b=3700  b=3800  b=3900  b=4000
## b=4100   b=4200  b=4300  b=4400  b=4500  b=4600  b=4700  b=4800  b=4900  b=5000
## b=5100   b=5200  b=5300  b=5400  b=5500  b=5600  b=5700  b=5800  b=5900  b=6000
## b=6100   b=6200  b=6300  b=6400  b=6500  b=6600  b=6700  b=6800  b=6900  b=7000
## b=7100   b=7200  b=7300  b=7400  b=7500  b=7600  b=7700  b=7800  b=7900  b=8000
## b=8100   b=8200  b=8300  b=8400  b=8500  b=8600  b=8700  b=8800  b=8900  b=9000
## b=9100   b=9200  b=9300  b=9400  b=9500  b=9600  b=9700  b=9800  b=9900  b=10000
```

```r
names(mt.test)
```

```
## [1] "index"    "teststat" "rawp"     "adjp"
```

```r
## $index : Vector of row indices, between 1 and nrow(X),
##          where rows are sorted first according to their adjusted p-values,
##          next their unadjusted p-values, and finally their test statistics.
## $rawp :  Vector of raw (unadjusted) p-values, ordered according to index.
## $adjp :  Vector of adjusted p-values, ordered according to index.

# Determine Euclidean dist. for each gene between sample classes
euc.dist <- function(x) sqrt( sum( (x[1:20] - x[21:40]) ^ 2 ) )
distance <- as.numeric(apply(genedata, 1, euc.dist))
summary(distance)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.438   5.703   6.451   6.809   7.321  13.700
```

```r
# Data frame shows the genes with an ascending order of adjusted p-values
diff.gene <- data.frame(distance=distance[mt.test$index], p.val=mt.test$rawp, adj.pval=mt.test$adjp)
head(diff.gene)
```

```
##   distance p.val adj.pval
## 1 12.18626 1e-04    1e-04
## 2 11.20657 1e-04    1e-04
## 3 13.46293 1e-04    1e-04
## 4 10.86794 1e-04    1e-04
## 5 12.43463 1e-04    1e-04
## 6 11.61549 1e-04    1e-04
```

```r
# Filter genes that have an adjusted p-value higher than 0.05
# i.e. differentially expressed genes
DEgenes <- diff.gene[which(diff.gene$adj.pval <= 0.05), ]
head(DEgenes)
```

```
##   distance p.val adj.pval
## 1 12.18626 1e-04    1e-04
```

```
## 2 11.20657 1e-04    1e-04
## 3 13.46293 1e-04    1e-04
## 4 10.86794 1e-04    1e-04
## 5 12.43463 1e-04    1e-04
## 6 11.61549 1e-04    1e-04
```

```r
# number of differentially expressed genes
nrow(DEgenes)
```

```
## [1] 105
```

```r
# Show differentially expressed genes with the highest distance
DEgenes <- DEgenes[order(DEgenes$distance, decreasing=T), ]

#most different 20 genes
DEgenes[1:20,]
```

```
##    distance p.val adj.pval
## 81 13.70364 1e-04   0.0018
## 32 13.61997 1e-04   0.0001
## 54 13.54408 1e-04   0.0003
## 43 13.49618 1e-04   0.0001
## 3  13.46293 1e-04   0.0001
## 46 13.39456 1e-04   0.0001
## 8  13.38039 1e-04   0.0001
## 68 13.24304 1e-04   0.0004
## 24 12.85060 1e-04   0.0001
## 16 12.81898 1e-04   0.0001
## 59 12.81057 1e-04   0.0004
## 42 12.71555 1e-04   0.0001
## 13 12.65287 1e-04   0.0001
## 33 12.48721 1e-04   0.0001
## 5  12.43463 1e-04   0.0001
## 7  12.33043 1e-04   0.0001
## 27 12.22267 1e-04   0.0001
## 1  12.18626 1e-04   0.0001
## 36 12.17640 1e-04   0.0001
## 26 12.15868 1e-04   0.0001
```

Appendix

R codes of the plots that were not presented in the text.

Figure 8

```
clusters <- cbind(km3$cluster, as.character(sim.data$Class))
cl1.3 <- which(clusters[,1] == 3 & clusters[,2] == "Class1")
cl2.1 <- which(clusters[,1] == 1 & clusters[,2] == "Class2")
cl3.2 <- which(clusters[,1] == 2 & clusters[,2] == "Class3")
cl3.3 <- which(clusters[,1] == 3 & clusters[,2] == "Class3")

par(mfrow=c(1,2), mar=c(4,4,4,1))

# plot the groups with V1 & 2
palette(c('#7fc97f','#beaed4','#fdc086'))
plot(sim.data[c("V1", "V2")], type="n", main="k=3 with V1 and V2")
text(sim.data[cl2.1, c("V1", "V2")], label=2, col='#beaed4', cex=2)
text(sim.data[cl3.2, c("V1", "V2")], label=3, col='#7fc97f', cex=2)
text(sim.data[cl3.3, c("V1", "V2")], label=3, col='#fdc086', cex=2)
text(sim.data[cl1.3, c("V1", "V2")], label=1, col='#fdc086', cex=2)
points(km3$centers[ , c("V1", "V2")], col = c('#7570b3','#1b9e77', '#d95f02'), pch = "*", cex = 5) #center
legend("bottomright", legend = paste("Cluster", 1:3), col=c('#beaed4','#7fc97f','#fdc086'), pch=20, pt.cex
legend("topright", legend = paste("Center", 1:3), col=c('#7570b3','#1b9e77', '#d95f02'), pch="*", pt.cex=3
# see which one was mis-clustered
points(sim.data[59, c("V1", "V2")]+0.1, col = '#e7298a', pch = "O", cex = 3)
text(sim.data[59, "V1"]+0.5, sim.data[59, "V2"]+1.9, col='#e7298a',
     labels="Misclassified \n Data Point", adj=c(0.6, 0.6), cex=1)

# plotting the PCA component 1 & 2
plot(pca.sim.data$x[,1:2], type="n", main="k=3 with PC1 and PC2")
text(pca.sim.data$x[cl2.1, 1:2], label=2, col='#beaed4', cex=2)
text(pca.sim.data$x[cl3.2, 1:2], label=3, col='#7fc97f', cex=2)
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], label=3, col='#fdc086', cex=2)
text(pca.sim.data$x[cl1.3, 1:2], label=1, col='#fdc086', cex=2)
legend("bottomright", legend = paste("Cluster", 1:3), col=c('#beaed4','#7fc97f','#fdc086'), pch=20, pt.cex
# see which one was mis-clustered
points(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], col = '#e7298a', pch = "O", cex = 3) #label=3, col='#
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2]+1.1, col='#e7298a',
     labels="Misclassified \n Data Point", adj=c(0.6, 0.6), cex=1)
```

Figure 9

```
clusters <- cbind(km2$cluster, as.character(sim.data$Class))
cl1 <- which(clusters[,1] == 1 & clusters[,2] == "Class1")
cl3 <- which(clusters[,1] == 1 & clusters[,2] == "Class3")
cl2 <- which(clusters[,1] == 2 & clusters[,2] == "Class2")

par(mfrow=c(1,2), mar=c(4,4,4,1))

#plot k=2 with V1 & V2
plot(sim.data[c("V1", "V2")], type='n', main="k=2 with V1 and V2")
```

```
text(sim.data[cl1, c("V1", "V2")], label=1, col='#7fc97f', cex=2)
text(sim.data[cl3, c("V1", "V2")], label=3, col='#7fc97f', cex=2)
text(sim.data[cl2, c("V1", "V2")], label=2, col='#beaed4', cex=2)
points(km2$centers[ , c("V1", "V2")], col = c('#1b9e77','#7570b3'), pch = "*", cex = 5) #center points of
legend("bottomright", legend = paste("Cluster", 1:2), col=c('#7fc97f','#beaed4'), pch=20, pt.cex=2, cex=1.
legend("topright", legend = paste("Center", 1:2), col=c('#1b9e77','#7570b3'), pch="*", pt.cex=3, cex=1.2)

# plotting the PCA component 1 & 2
plot(pca.sim.data$x[,1:2], type="n", main="k=3 with PC1 and PC2")
text(pca.sim.data$x[cl2, 1:2], label=2, col='#beaed4', cex=2)
text(pca.sim.data$x[cl3, 1:2], label=3, col='#7fc97f', cex=2)
text(pca.sim.data$x[cl1, 1:2], label=1, col='#7fc97f', cex=2)
legend("bottomright", legend = paste("Cluster", 1:2), col=c('#beaed4','#7fc97f'), pch=20, pt.cex=2, cex=1.
```

Figure 10

```
clusters <- cbind(km4$cluster, as.character(sim.data$Class))
cl1.4 <- which(clusters[,1] == 4 & clusters[,2] == "Class1")
cl2.1 <- which(clusters[,1] == 1 & clusters[,2] == "Class2")
cl3.2 <- which(clusters[,1] == 2 & clusters[,2] == "Class3")
cl3.3 <- which(clusters[,1] == 3 & clusters[,2] == "Class3")
cl3.4 <- which(clusters[,1] == 4 & clusters[,2] == "Class3")

# ['#1b9e77', '#7570b3','#d95f02','#386cb0']
# ['#7fc97f','#beaed4','#fdc086','#80b1d3']

par(mfrow=c(1,2), mar=c(4,4,4,1))

plot(sim.data[c("V1", "V2")], type='n', main="k=4 with V1 and V2")
text(sim.data[cl1.4, c("V1", "V2")], label=1, col='#80b1d3', cex=2) #cl4
text(sim.data[cl2.1, c("V1", "V2")], label=2, col='#beaed4', cex=2) #cl1
text(sim.data[cl3.2, c("V1", "V2")], label=3, col='#7fc97f', cex=2) #cl2
text(sim.data[cl3.3, c("V1", "V2")], label=3, col='#fdc086', cex=2) #cl3
text(sim.data[cl3.4, c("V1", "V2")], label=3, col='#80b1d3', cex=2) #cl4

points(km4$centers[ , c("V1", "V2")], col = c('#7570b3','#1b9e77', '#d95f02','#386cb0'), pch = "*", cex =
legend("bottomright", legend = paste("Cluster", 1:4), col=c('#beaed4','#7fc97f','#fdc086','#80b1d3'), pch=
legend("topright", legend = paste("Center", 1:4), col=c('#7570b3','#1b9e77', '#d95f02','#386cb0'), pch="*"
points(sim.data[59, c("V1", "V2")]+0.1, col = '#e7298a', pch = "O", cex = 3)
text(sim.data[59, "V1"]+0.5, sim.data[59, "V2"]+1.3, col='#e7298a',
     labels="Observation 59", adj=c(0.6, 0.6), cex=1)
points(sim.data[59, c("V1", "V2")]+0.1, col = '#e7298a', pch = "O", cex = 3)


# plotting the PCA component 1 & 2
plot(pca.sim.data$x[,1:2], type="n", main="k=3 with PC1 and PC2")
text(pca.sim.data$x[cl1.4, 1:2], label=1, col='#80b1d3', cex=2) #cl4
text(pca.sim.data$x[cl2.1, 1:2], label=2, col='#beaed4', cex=2) #cl1
text(pca.sim.data$x[cl3.2, 1:2], label=3, col='#7fc97f', cex=2) #cl2
text(pca.sim.data$x[cl3.3, 1:2], label=3, col='#fdc086', cex=2) #cl3
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], label=3, col='#80b1d3', cex=2) #cl4
legend("bottomright", legend = paste("Cluster", 1:4), col=c('#beaed4','#7fc97f','#fdc086','#80b1d3'), pch=

points(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2], col = '#e7298a', pch = "O", cex = 3) #label=3, col='#
```

```
text(pca.sim.data$x[59, 1], pca.sim.data$x[59, 2]+1.1, col='#e7298a',
     labels="Observation 59", adj=c(0.6, 0.6), cex=1)
```

Figure 11

```
library(dendextend)

groupCodes <- c(rep("H",20), rep("D",20))
colorCodes <- c(H="#2c7bb6", D='#d7191c')
linkage <- c( "single", "complete", "average", "centroid")
par(mfrow=c(2,2), mar=c(3,2,3,2))
for(i in 1:length(linkage)){
  d <- as.dist((1 - cor(genedata))/2) #correlation based distance as suggested in ?dist help file
  hc <- hclust(d, method=linkage[i])            # apply hirarchical clustering with different linkages
  hcd<-as.dendrogram(hc, dLeaf=0.1, h=0.3)


  # Assigning the labels of dendrogram object with new colors:
  labels_colors(hcd) <- colorCodes[groupCodes][order.dendrogram(hcd)]
  # Plotting the new dendrogram
  plot(hcd, main=paste(linkage[i], "linkage", sep=" "), ylim=c(0.1 , 0.6))

}
```