Correlation Based Principal Component Analysis

P. Anipa

Reading and viewing the Data

We wish to use the data without the variables Points, Height and Weight. So we remove them from the data.

```
decat = read.table("decathlon.txt", header = TRUE, sep = "\t", row.names = 1)
head(decat)
```

```
##          Points R100m Long_jump Shot_put High_jump R400m Hurdles Discus_throw
## Skowrone   8206   853       931      725       857   838     903          772
## Hedmark    8188   853       853      814       769   833     914          855
## Le_Roy     8140   879       951      799       779   838     881          819
## Zeilbaue   8136   826       931      793       865   875     891          729
## Zigert     8134   879       840      924       857   788     892          866
## Bennett    8121   905       859      647       779   938     859          651
##          Pole_vault Javelin R1500m Height Weight
## Skowrone        981     818    528    184     81
## Hedmark         884     975    438    195     90
## Le_Roy         1028     758    408    191     90
## Zeilbaue        909     774    543    192     84
## Zigert          920     671    497    198    105
## Bennett        1028     794    661    173     68
```

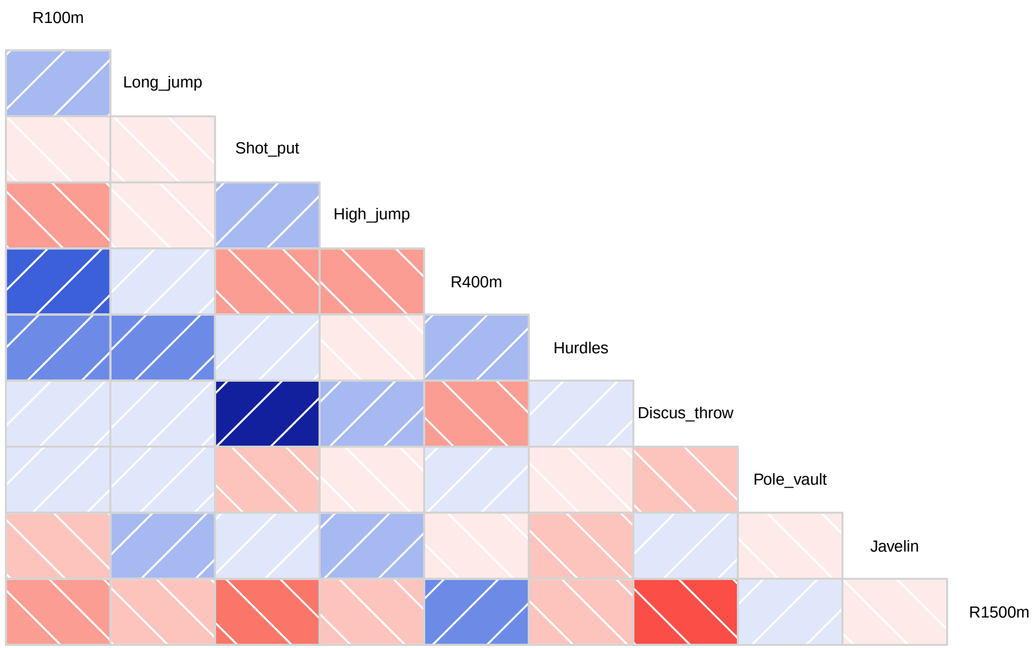
```
decat_r = decat[ , -c(1, 12, 13)]
head(decat_r)
```

```
##          R100m Long_jump Shot_put High_jump R400m Hurdles Discus_throw
## Skowrone   853       931      725       857   838     903          772
## Hedmark    853       853      814       769   833     914          855
## Le_Roy     879       951      799       779   838     881          819
## Zeilbaue   826       931      793       865   875     891          729
## Zigert     879       840      924       857   788     892          866
## Bennett    905       859      647       779   938     859          651
##          Pole_vault Javelin R1500m
## Skowrone        981     818    528
## Hedmark         884     975    438
## Le_Roy         1028     758    408
## Zeilbaue        909     774    543
## Zigert          920     671    497
## Bennett        1028     794    661
```

Visualizing the correlation matrix

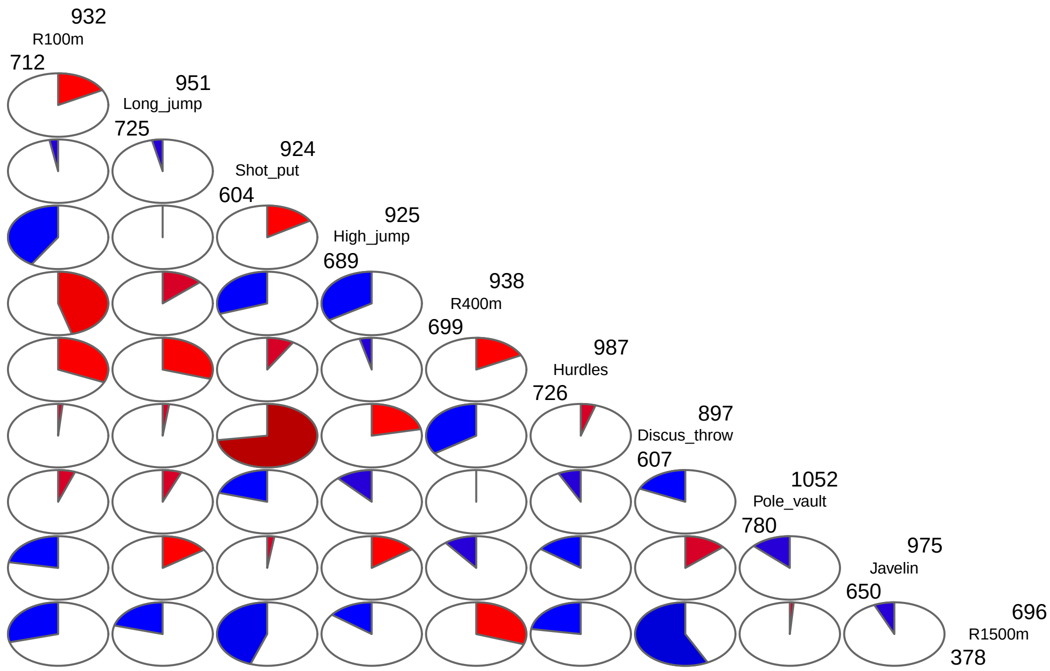
The correlation matrix can be visualized with a heat map. In this plot blue indicates positive correlation and red indicates negative correlation.

```r
#install.packages("corrgram")
library(corrgram)
corrgram(decat_r, upper.panel = NULL)
```



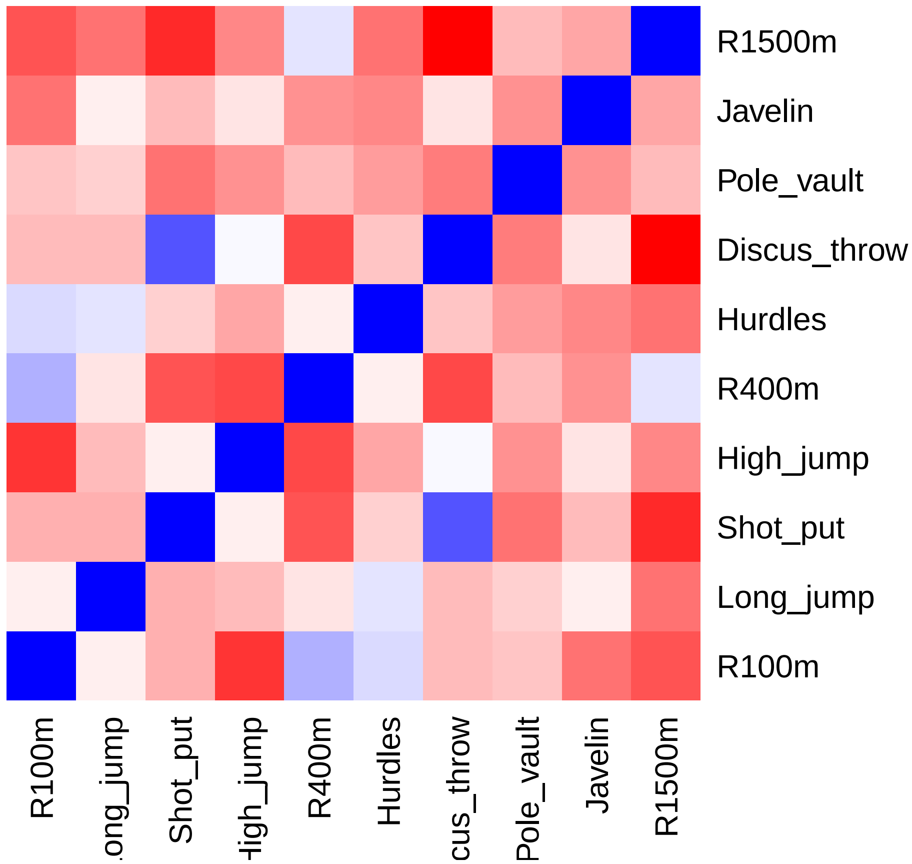
A pie chart can be used to visualize the correlation matrix. In this plot red indicates positive correlation and blue indicates negative correlation.

```r
colours = c("blue4","blue3", "blue2", "blue1", "blue", "red", "red1",  "red2",  "red3", "red4")
corrgram(decat_r, lower.panel = panel.pie, diag.panel = panel.minmax, upper.panel = NULL, col.regions =
```



Base R can also be used for plotting a heatmap

```
heatmap(cor(decat_r), Rowv = NA, Colv = NA, symm = TRUE, col = colorRampPalette(c("red", "white", "blue
```



Correlation matrix based PCA

We now perform the correlation matrix based principal component analysis.(We make sure to set 'cor' to true)

```
decat_pca = princomp(decat_r, cor = TRUE)
summary(decat_pca)
```

```
## Importance of components:
##                           Comp.1    Comp.2   Comp.3    Comp.4     Comp.5
## Standard deviation     1.6130891 1.4169733 1.098463 1.0329820 0.96516226
## Proportion of Variance 0.2602056 0.2007813 0.120662 0.1067052 0.09315382
## Cumulative Proportion  0.2602056 0.4609870 0.581649 0.6883542 0.78150800
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     0.77116160 0.75437360 0.73395022 0.49482809 0.48745515
## Proportion of Variance 0.05946902 0.05690795 0.05386829 0.02448548 0.02376125
## Cumulative Proportion  0.84097702 0.89788497 0.95175326 0.97623875 1.00000000
```

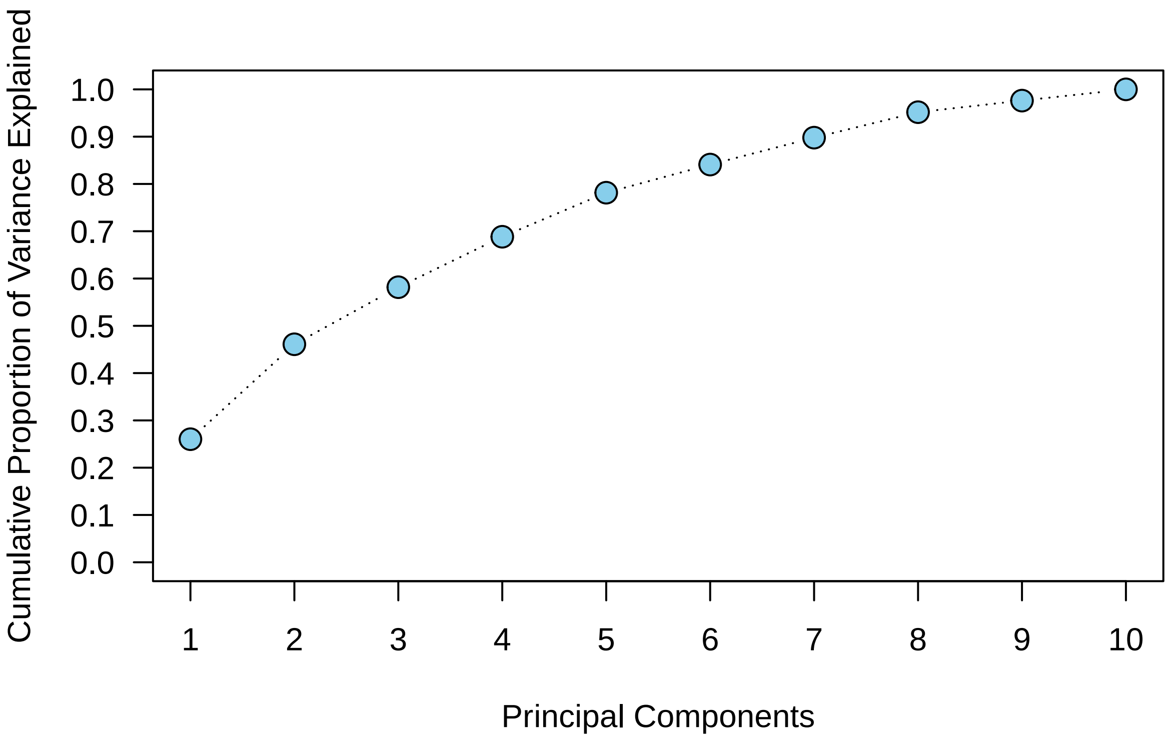
How much variation is explained by k principal components?

We want to find out how much variation is explained by k principal components. The summary tells us this information in the row "Proportion of Variance". We can also show a visualization of how much variation is explained by k principal components.

```
var = decat_pca$sdev^2
var_prop = var / sum(var) #Proportion of variance
var_prop_cum = cumsum(var_prop) #Cumulative proportion
```

```r
plot(var_prop_cum, type = "b", pch = 21, lty = 3, bg = "skyblue", cex = 1.5,
     ylim = c(0, 1), xlab = "Principal Components",
     ylab = "Cumulative Proportion of Variance Explained",
     xaxt = "n", yaxt = "n")

axis(1, at = 1:10)
axis(2, at = 0:10/10, las = 2)
```

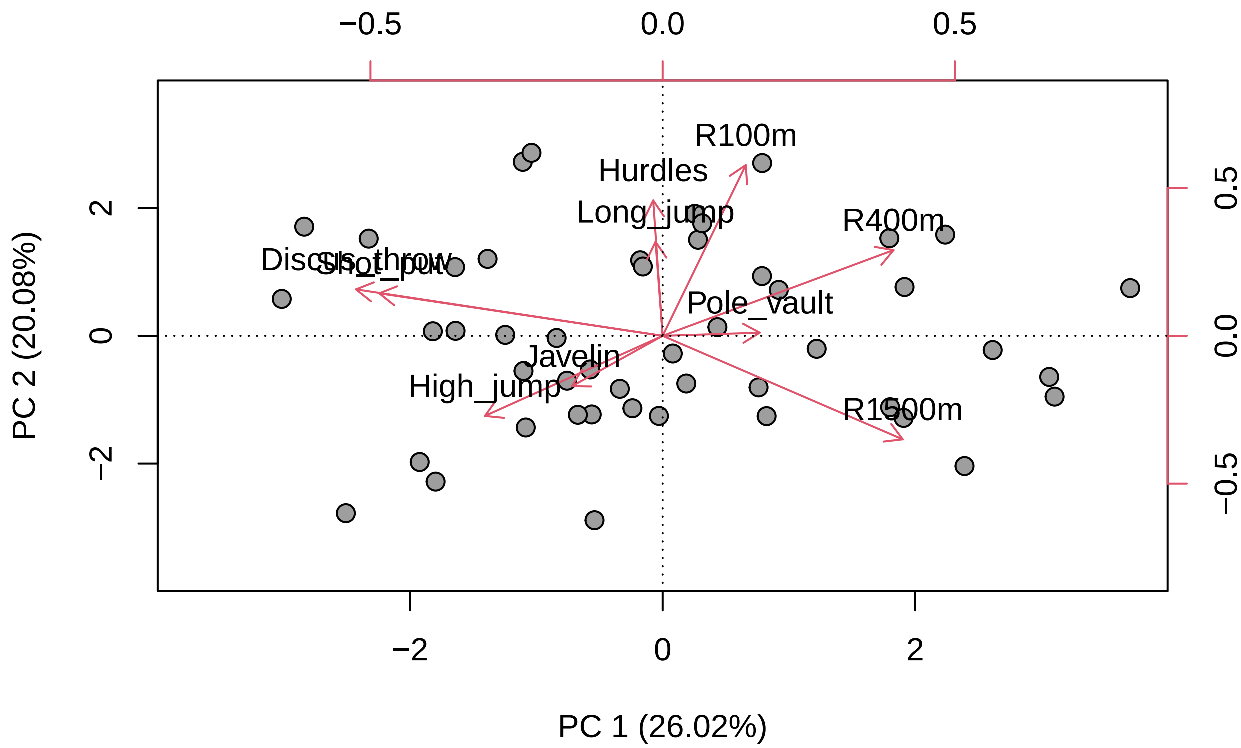


Interpretation of principal components

We wish to interpret the first four principal components because combined, they represent close to 70% of the variance. We first show a Biplot of the scores and loadings.

```r
pc12 <- decat_pca$scores[, 1:2]
load12 <- decat_pca$loadings[, 1:2]
pc_axis <- c(-max(abs(pc12)), max(abs(pc12)))
ld_axis <- c(-0.8, 0.8)
plot(pc12, xlim = pc_axis, ylim = pc_axis, pch = 21, bg = 8, cex = 1.25,
     xlab = paste0("PC 1 (", round(100 * var_prop[1], 2), "%)"),
     ylab = paste0("PC 2 (", round(100 * var_prop[2], 2), "%)"))
par(new = TRUE)
plot(load12, axes = FALSE, type = "n", xlab = "", ylab = "", xlim = ld_axis,
     ylim = ld_axis)
axis(3, col = 2)
axis(4, col = 2)
arrows(0, 0, load12[, 1], load12[, 2], length = 0.1, col = 2)
```

```r
text(load12[, 1], load12[, 2], rownames(load12), pos = 3)
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
```



We can interpret the principal components by looking at the loadings.

```r
round(decat_pca$loadings[, 1:4], 2)
```

```
##              Comp.1 Comp.2 Comp.3 Comp.4
## R100m          0.14   0.58   0.15   0.03
## Long_jump     -0.01   0.32  -0.65  -0.21
## Shot_put      -0.48   0.14   0.24   0.13
## High_jump     -0.30  -0.27  -0.27  -0.07
## R400m          0.40   0.29  -0.08   0.32
## Hurdles       -0.02   0.46  -0.19   0.07
## Discus_throw  -0.52   0.16   0.14   0.05
## Pole_vault     0.17   0.01   0.08  -0.86
## Javelin       -0.16  -0.17  -0.60   0.15
## R1500m         0.41  -0.35   0.00   0.25
```

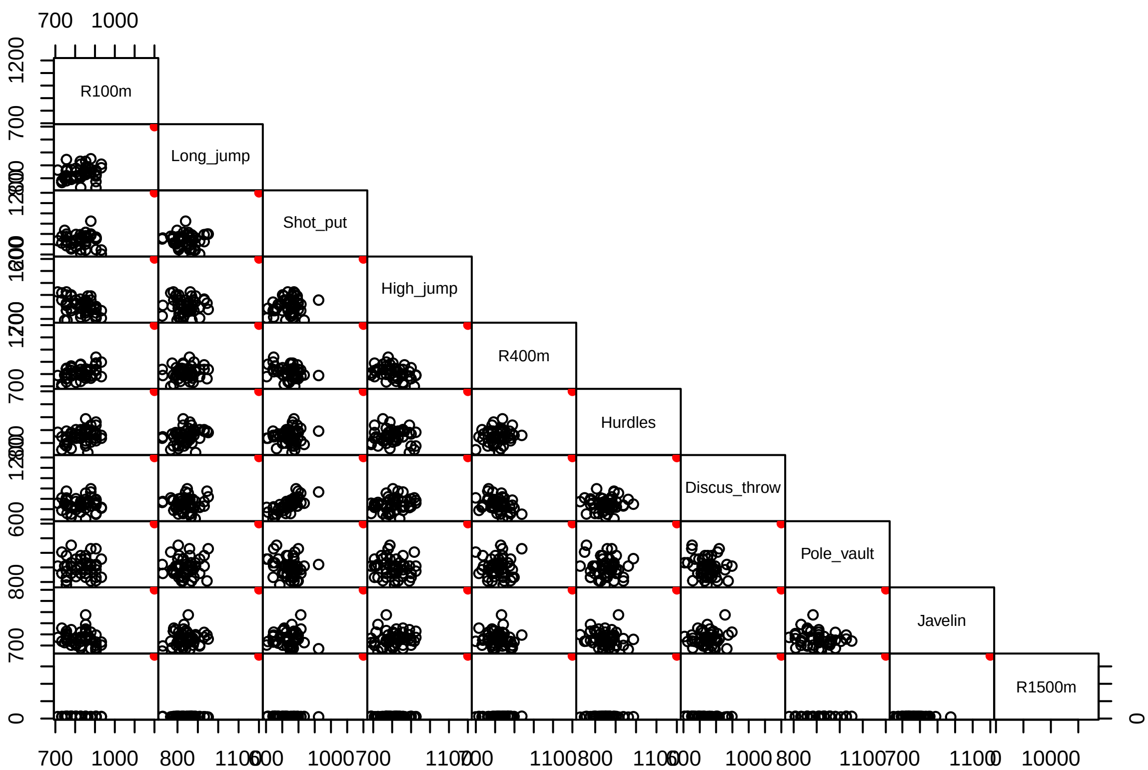
Example: Possible interpretation for the first component is strength.

Outlier detection by principal components

We wish to add one clear outlier into the data set and use PCA to detect the outlier.

```r
q = 9
decat_r[49, ] = c(rep(1200, q), rep(18000, 10 - q))
rownames(decat_r)[49] = "outlier"

#Visualize the new data which contains the outlier using a pairwise scatter plot
pairs(decat_r, gap = 0, upper.panel = NULL, pch = c(rep(1, 48), 16),
      col = c(rep("black", 48), "red"))
```



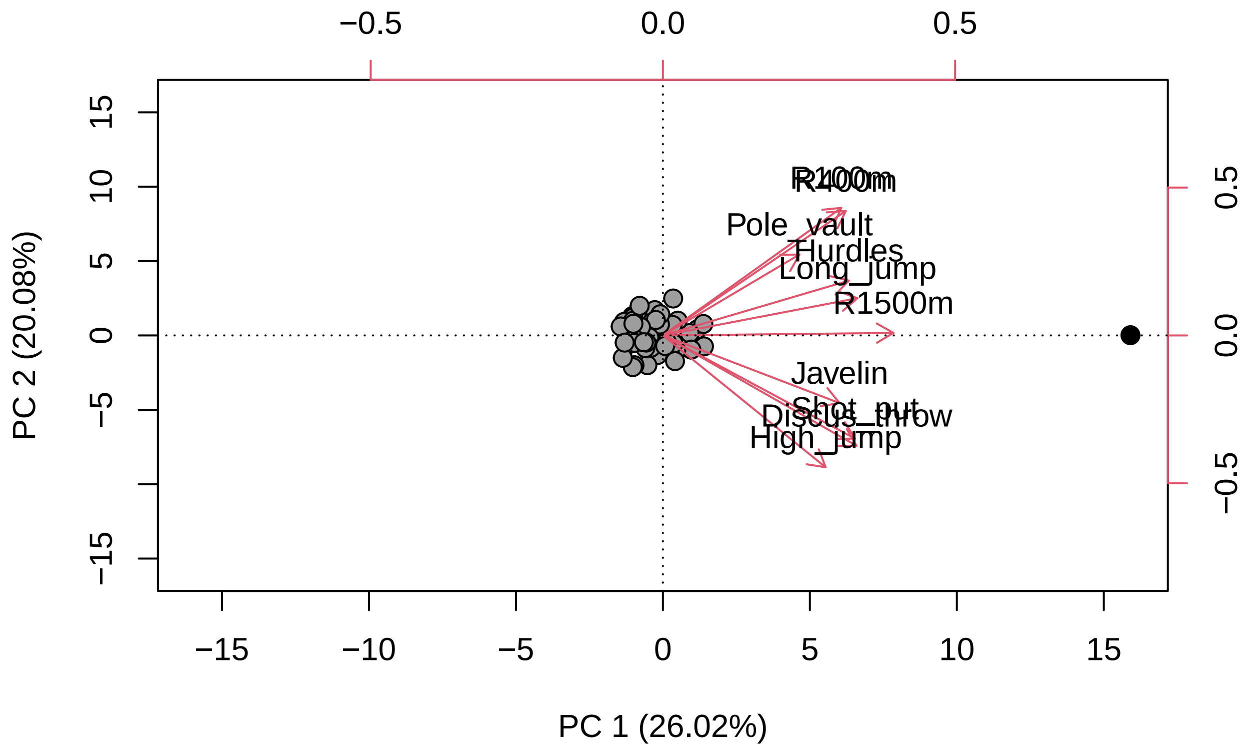
Perform PCA on ‘contaminated’ data.

```r
decat_pca_b = princomp(decat_r, cor = TRUE)
vars = decat_pca$sdev^2
var_prop = vars / sum(vars)


pc12 = decat_pca_b$scores[, 1:2]
load12 = decat_pca_b$loadings[, 1:2]
pc_axis = c(-max(abs(pc12)), max(abs(pc12)))
ld_axis = c(-0.8, 0.8)

plot(pc12, xlim = pc_axis, ylim = pc_axis, pch = 21, bg = c(rep(8, 48), 1),
     cex = 1.25, xlab = paste0("PC 1 (", round(100 * var_prop[1], 2), "%)"),
     ylab = paste0("PC 2 (", round(100 * var_prop[2], 2), "%)"))
par(new = TRUE)
plot(load12, axes = FALSE, type = "n", xlab = "", ylab = "", xlim = ld_axis,
     ylim = ld_axis)
```

```r
axis(3, col = 2)
axis(4, col = 2)
arrows(0, 0, load12[, 1], load12[, 2], length = 0.1, col = 2)
text(load12[, 1], load12[, 2], rownames(load12), pos = 3)
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
```



From the biplot we can see that the first principal component detects outlier very well. On the other hand, the outlier is not as well detected by the 2nd or 3rd principal components as shown below.

Also, it can be seen that PCA is quite a nonrobust method. That is, outliers have significant effect to the results of PCA.

```r
pc23 = decat_pca_b$scores[, 2:3]
load23 = decat_pca_b$loadings[, 2:3]
pc_axis = c(-max(abs(pc23)), max(abs(pc23)))
ld_axis = c(-0.8, 0.8)

plot(pc23, xlim = pc_axis, ylim = pc_axis, pch = 21, bg = c(rep(8, 48), 1),
     cex = 1.25, xlab = paste0("PC 2 (", round(100 * var_prop[2], 2), "%)"),
     ylab = paste0("PC 3 (", round(100 * var_prop[3], 2), "%)"))
par(new = TRUE)
plot(load23, axes = FALSE, type = "n", xlab = "", ylab = "", xlim = ld_axis,
     ylim = ld_axis)
axis(3, col = 2)
axis(4, col = 2)
arrows(0, 0, load23[, 1], load23[, 2], length = 0.1, col = 2)
```

```r
text(load23[, 1], load23[, 2], rownames(load23), pos = 3)
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
```