Basic PCAs

Clarke van Steenderen

2025-02

Running a Principal Components Analysis (PCA) in R

A PCA is a statistical method that simplifies large data sets with multiple variables by reducing their dimensionality. It captures important trends and patterns in the data, and is easy to visualise and interpret. Imagine we are investigating which variables have an effect on life expectancy. If we were only looking at two variables, perhaps cholesterol level and body mass index, it is quite simple to plot these against each other, and look for a correlation. But what if we have another ten or more variables to consider? Whether a person smokes or not, blood pressure, exercise regularity, height, etc. Plotting all these in various combinations can become near impossible. This is where a PCA comes in handy, as it reduces all these variables into something that is much easier to work with and interpret.

There are a number of steps in the process, which include standardisation of the data (variables may have differing scales), covariance calculations (checking the relationships between the different variables), and determining Eigen vectors and eigenvalues. These values are then re-projected onto a new coordinate system.

```r
# install the required packages, if not available already
if (!require("pacman")) install.packages("pacman")
pacman::p_load(tidyverse,
               tidyr,
               janitor,
               vegan,
               readr,
               magrittr,
               factoextra,
               ggpubr,
               ggplot2,
               rgl)

biopsy_data = MASS::biopsy

# Columns refer to these nine attributes:
colnames(biopsy_data) <- c(
   "id",
   "clump_thickness",
   "uniform_cell_size" ,
   "uniform_cell_shape",
   "marg_adhesion",
   "epithelial_cell_size",
   "bare_nuclei",
   "bland_chromatin",
   "normal_nucleoli",
   "mitoses",
   "outcome"
)

# Let's first check whether there are any NA values
any(is.na(biopsy_data))
```

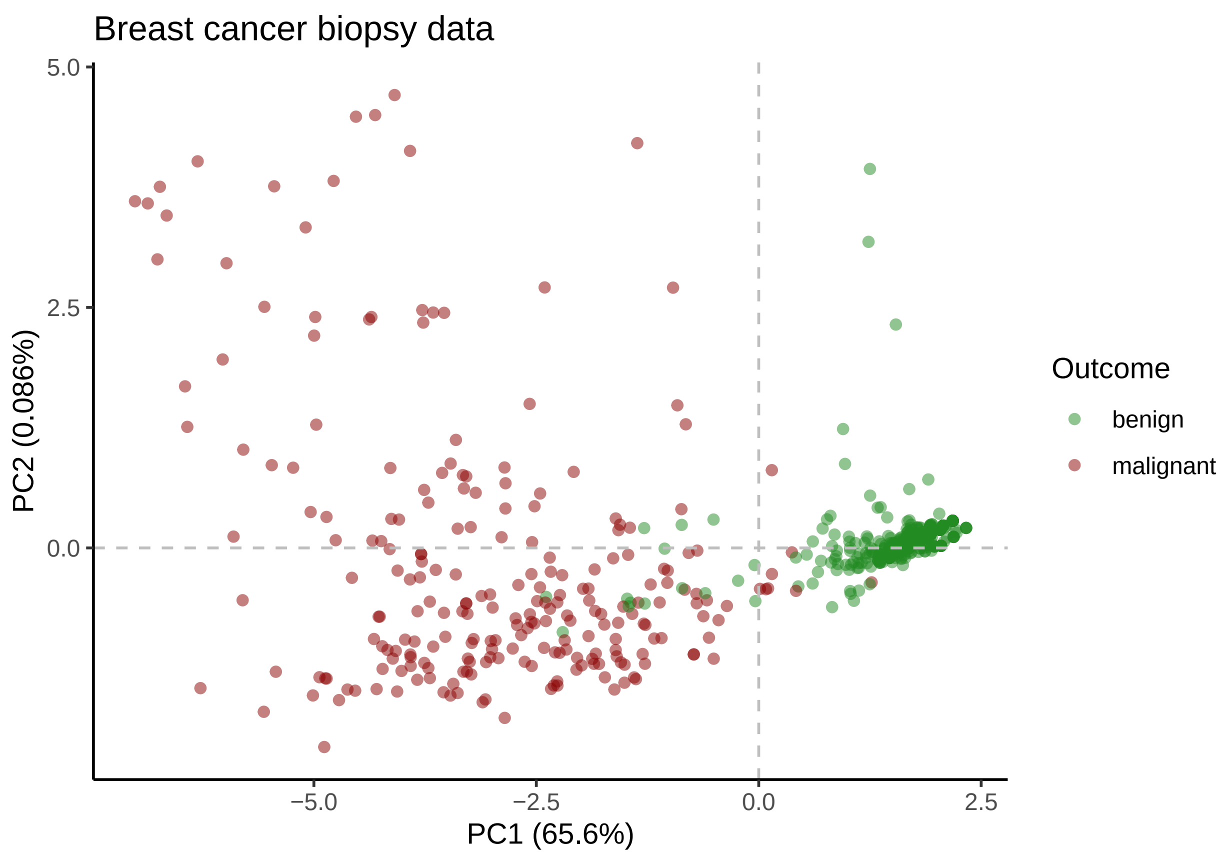
```
## [1] TRUE
```

```r
biopsy_data = na.omit(biopsy_data)

# since there is at least one non-numeric column in the
# dataframe (id and outcome), let's remove
biopsy_data_clean = biopsy_data %>%
  dplyr::select(where(is.numeric))

pca_fit <- biopsy_data_clean %>%
  prcomp(scale = TRUE) # scale the data before doing the PCA

# PCA summary results
summary(pca_fit)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4289 0.88088 0.73434 0.67796 0.61667 0.54943 0.54259
## Proportion of Variance 0.6555 0.08622 0.05992 0.05107 0.04225 0.03354 0.03271
## Cumulative Proportion  0.6555 0.74172 0.80163 0.85270 0.89496 0.92850 0.96121
##                            PC8     PC9
## Standard deviation     0.51062 0.29729
## Proportion of Variance 0.02897 0.00982
## Cumulative Proportion  0.99018 1.00000
```

```r
# PC1: 65.6% variation
# PC2: 8.6% variation

# Let's have a look at the loadings
pca_loadings = pca_fit$rotation %>%
  as.data.frame() %>%
  dplyr::select(c(PC1, PC2))

# Let's get all the PC scores
pca_fit_df = pca_fit$x %>%
  as.data.frame() %>%
  # keep only PC1 and PC2
  dplyr::select(c(PC1, PC2))

# Add outcome column back in
# adding a dollar sign with a new name tells R to append a column to the end
# here, it will create a column called "outcome" to the end of pca_fit_df
pca_fit_df$outcome = biopsy_data$outcome

# change the alpha value to add transparency to the colours
```

```r
pca_plot = pca_fit_df %>%
  ggplot(aes(PC1, PC2, color = outcome)) +
  geom_point(size = 1.5, alpha = 0.5) +
  scale_color_manual(
    values = c(malignant = "darkred", benign = "forestgreen")
  ) +
  xlab("PC1 (65.6%)") +
  ylab("PC2 (0.086%)") +
  labs(color = "Outcome",
       title = "Breast cancer biopsy data") +
  # add lines through the origin
  geom_hline(yintercept = 0, linetype = "dashed", color = "gray") +
  geom_vline(xintercept = 0, linetype = "dashed", color = "gray") +
  theme_classic()

pca_plot
```



We can tell that 66% of the variation in the data is captured on PC1, which is also shown clearly by the grouping of most of the "benign" points on the positive side of the x-axis/PC1. The largest loading on PC1 was uniform cell size, at -0.38. All the negative loading values for PC1 just indicates that each variable increases along the negative axis. For example, clump thickness increases along the negative side of the x-axis -> in the direction of the dark red malignant points on the PCA plot.

In addition to **prcomp()**, one can also use **princomp()**. The two methods take slighlty different statistical approaches, but should show comparable outputs. Let's have a look at how one would apply the **princomp()** method, with some biplots:

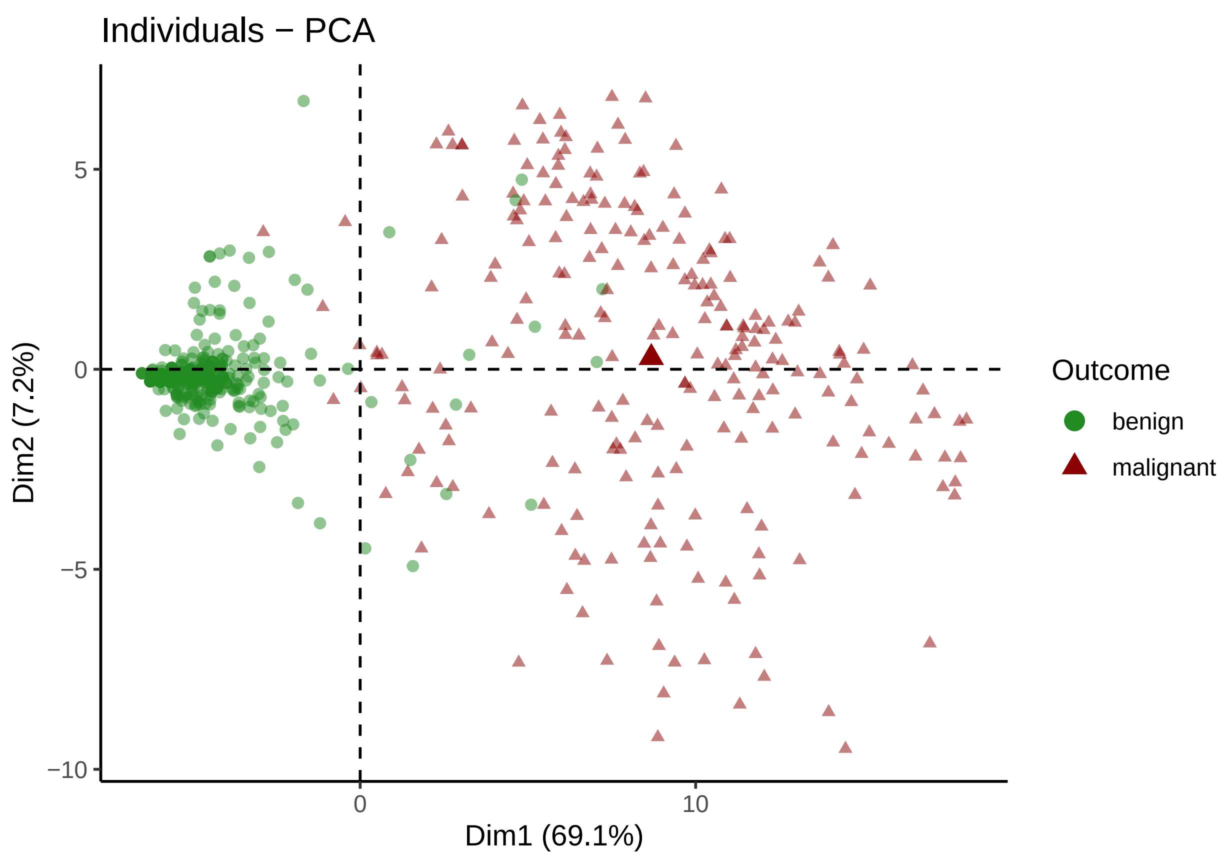
```r
# We can also use the princomp() function to run a PCA
princomp_pca = princomp(biopsy_data_clean)

# Here, PC1 variance is 69% and PC2 is 0.072
summary(princomp_pca)
```

```
## Importance of components:
##                           Comp.1     Comp.2     Comp.3     Comp.4     Comp.5
## Standard deviation     6.9982536 2.25903450 2.07250492 1.77498764 1.66328370
## Proportion of Variance 0.6905076 0.07195066 0.06055921 0.04442012 0.03900513
## Cumulative Proportion  0.6905076 0.76245823 0.82301744 0.86743756 0.90644270
##                            Comp.6     Comp.7     Comp.8     Comp.9
## Standard deviation     1.56289816 1.33943265 1.26229092 0.89756328
## Proportion of Variance 0.03443899 0.02529478 0.02246508 0.01135845
## Cumulative Proportion  0.94088169 0.96617646 0.98864155 1.00000000
```

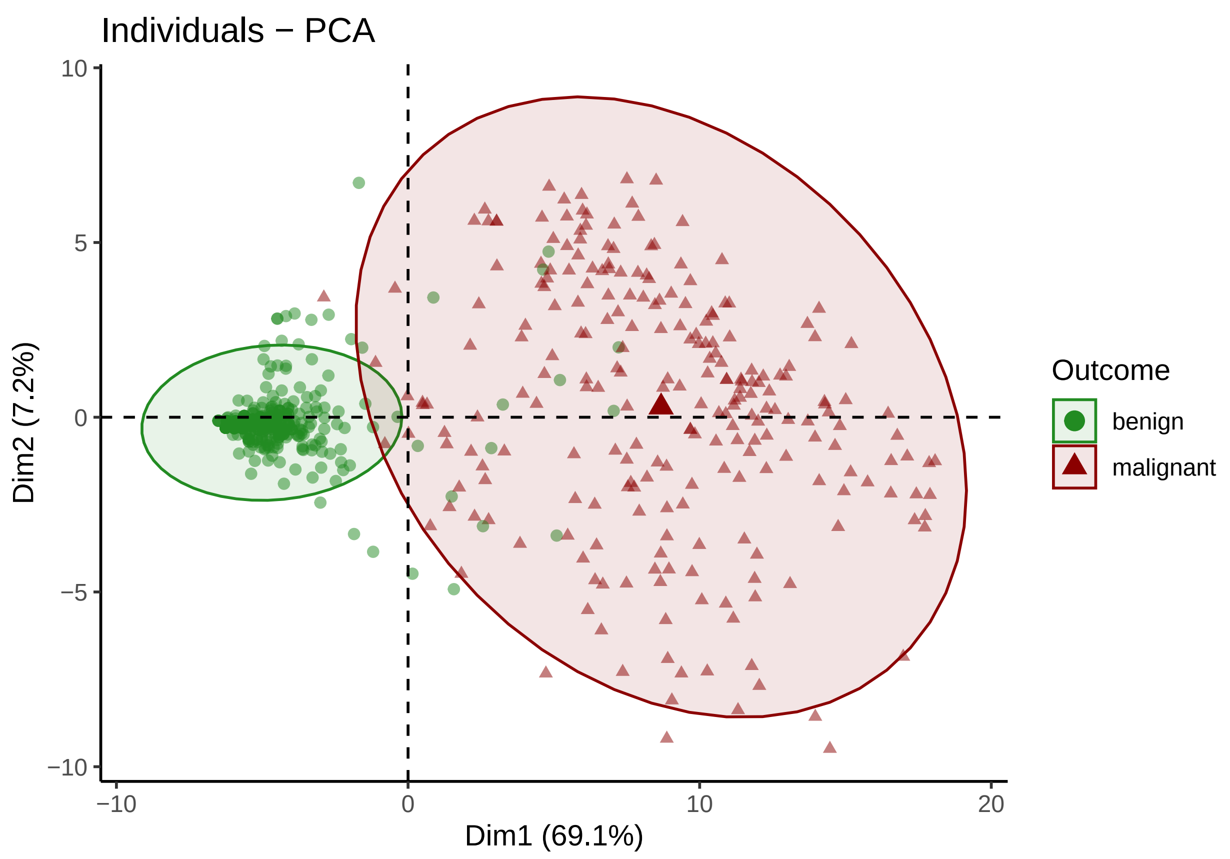
```r
# loadings for the first two components.
# note the difference in sign -> these are all positive
princomp_pca$loadings[, 1:2]
```

```
##                         Comp.1       Comp.2
## clump_thickness      0.2967358  0.073506644
## uniform_cell_size    0.4039707 -0.229928848
## uniform_cell_shape   0.3927586 -0.164700982
## marg_adhesion        0.3312021  0.098197542
## epithelial_cell_size 0.2497398 -0.200215050
## bare_nuclei          0.4426135  0.780569633
## bland_chromatin      0.2920783 -0.008479735
## normal_nucleoli      0.3545360 -0.469194517
## mitoses              0.1245763 -0.188068892
```

```r
# plot the PCA. What's nice about princomp, is that you can add ggplot features
factoextra::fviz_pca_ind(princomp_pca, geom = "point",
                         col.ind = biopsy_data$outcome,
                         palette = c("forestgreen",  "darkred"),
                         legend.title = "Outcome", alpha.ind = 0.5) +
  scale_shape_manual(values=c(19,17)) +
  theme_classic()
```



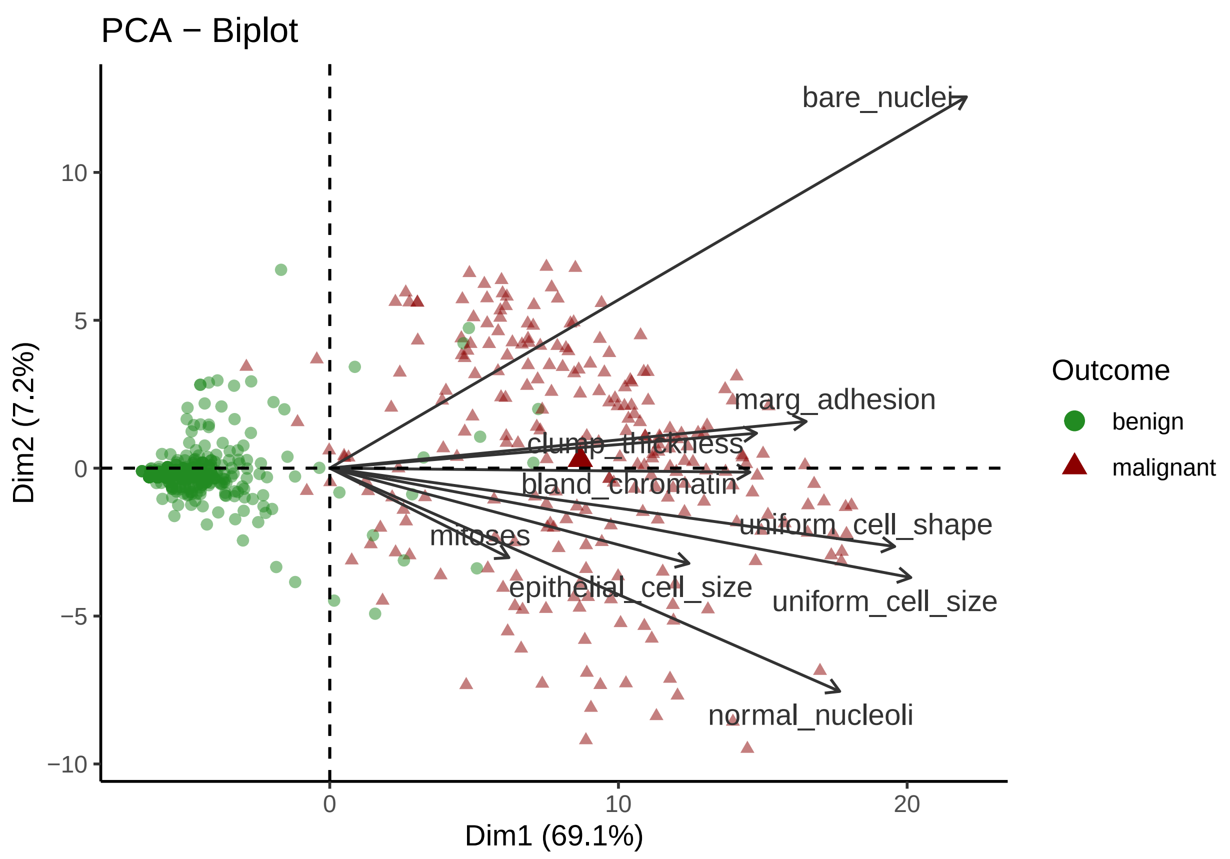
```r
# Have a look at what it looks like when we add ellipses around groups
factoextra::fviz_pca_ind(princomp_pca, geom = "point",
                         col.ind = biopsy_data$outcome,
                         palette = c("forestgreen",  "darkred"),
                         legend.title = "Outcome", alpha.ind = 0.5,
                         addEllipses = TRUE) +
  scale_shape_manual(values=c(19,17)) +
  theme_classic()
```



You can see that the benign data points have lower PC1 values, and cluster closely together.

Let's have a look at a PCA biplot. Biplots show all the PCA scores, with the loading vectors as arrows. The length of these arrows indicates the strength and direction of the loading of each variable on PC1 and PC2.

```r
factoextra::fviz_pca_biplot(princomp_pca, repel = TRUE,
                            geom = "point",
                            col.ind = biopsy_data$outcome,
                            palette = c("forestgreen",  "darkred"),
                            alpha.ind = 0.5,
                            legend.title = "Outcome",
                            col.var = "grey20" ) +
  scale_shape_manual(values=c(19,17)) +
  theme_classic()
```



In this biplot, we can see that all the variables increase on the positive side of PC1, and are all positively correlated with each other. We can infer that an increase in all these variables is associated with a higher chance of malignancy.

An example with the Iris data set

Let's load in the Iris data again, and run a PCA that accounts for sepal and petal width and length across the three species.

```r
# access the embedded dataset
iris_data = datasets::iris
iris_data = janitor::clean_names(iris_data)

head(iris_data)
```

```
##   sepal_length sepal_width petal_length petal_width species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
str(iris_data)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ sepal_length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ sepal_width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ petal_length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ petal_width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
# Let's first check whether there are any NA values
any(is.na(iris_data))
```

```
## [1] FALSE
```

```r
# since there is a non-numeric column (species) in the dataframe,
# let's remove it
iris_data_clean = iris_data %>%
  dplyr::select(where(is.numeric))

pca_iris = prcomp(iris_data_clean, scale = TRUE)

# Here, PC1 variance is 73% and PC2 is 23%
summary(pca_iris)
```

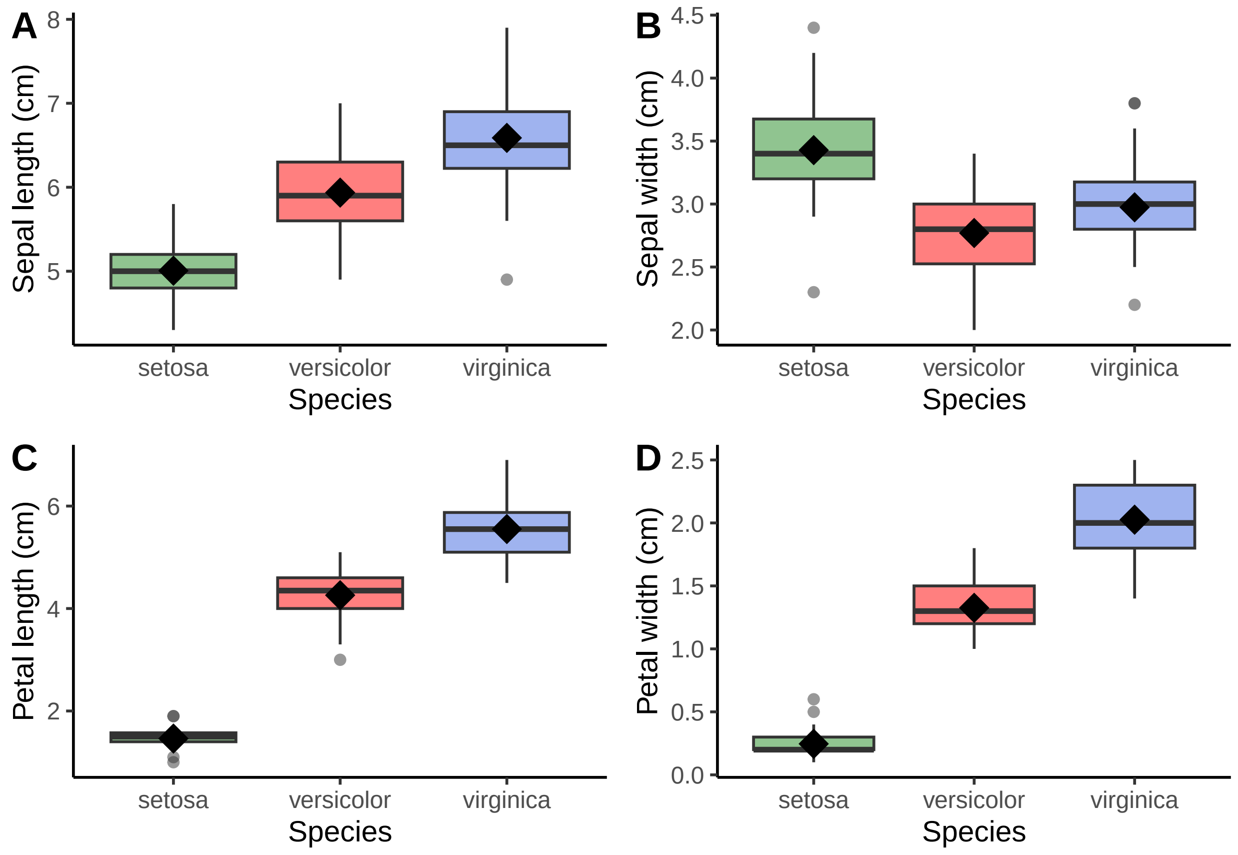
```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7084 0.9560 0.38309 0.14393
## Proportion of Variance 0.7296 0.2285 0.03669 0.00518
## Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
```

```r
# Let's have a look at the loadings
pca_loadings_iris = pca_iris$rotation %>%
  as.data.frame() %>%
  dplyr::select(c(PC1, PC2))

# Let's get all the PC scores
pca_iris_df = pca_iris$x %>%
  as.data.frame() %>%
  # keep only PC1 and PC2
  dplyr::select(c(PC1, PC2))
```

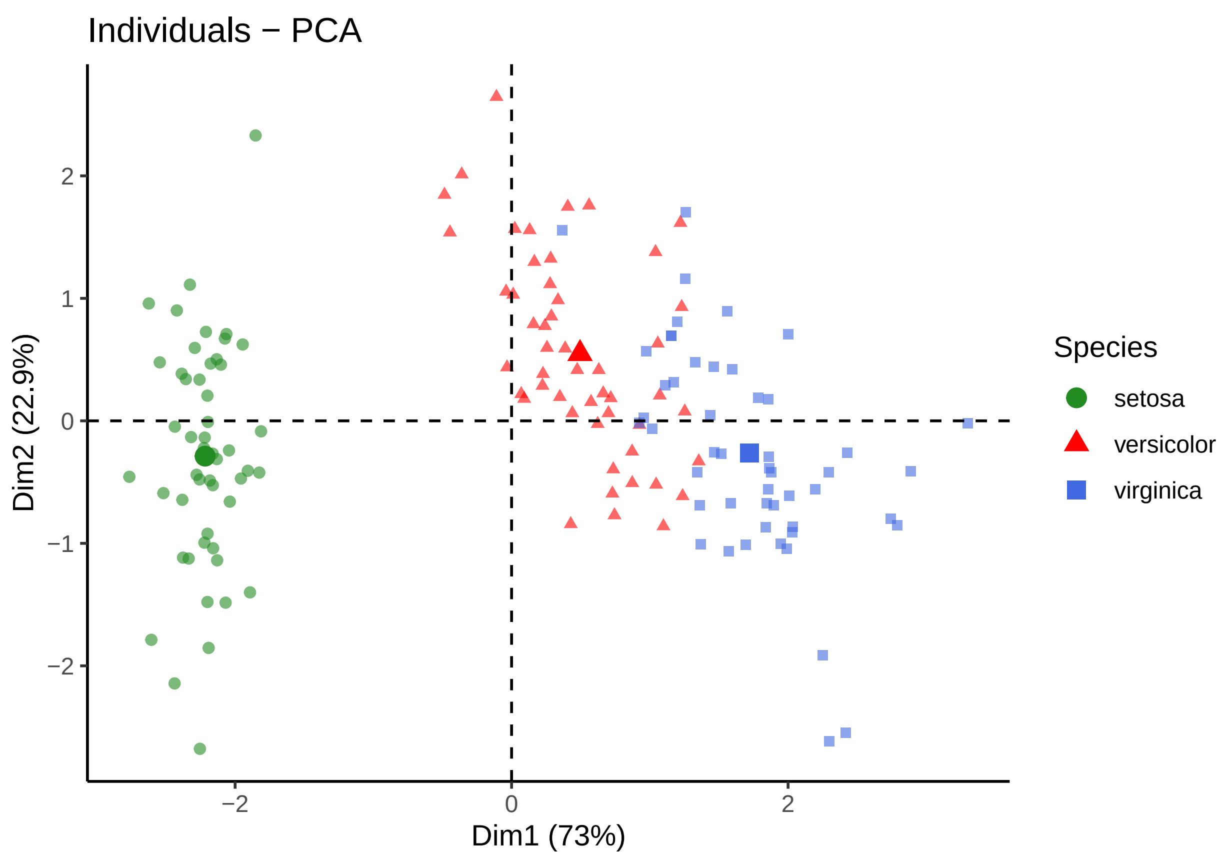
These loadings tell us that all the measurements except sepal width increase along the positive side of PC1 (the x axis). Sepal width increases along the negative side of the x axis/PC2. When we plot these, we can see how *setosa* separates out quite clearly (green) along PC1, and we can infer from this that it has **shorter** sepal lengths, petal widths, and petal lengths than the other species, but **larger** sepal widths. The magnitude of the PC scores indicate how important they are, and are shown in the biplot by the length of the arrows.

```r
# let's check these patterns using a box plots

# SEPAL LENGTH
```

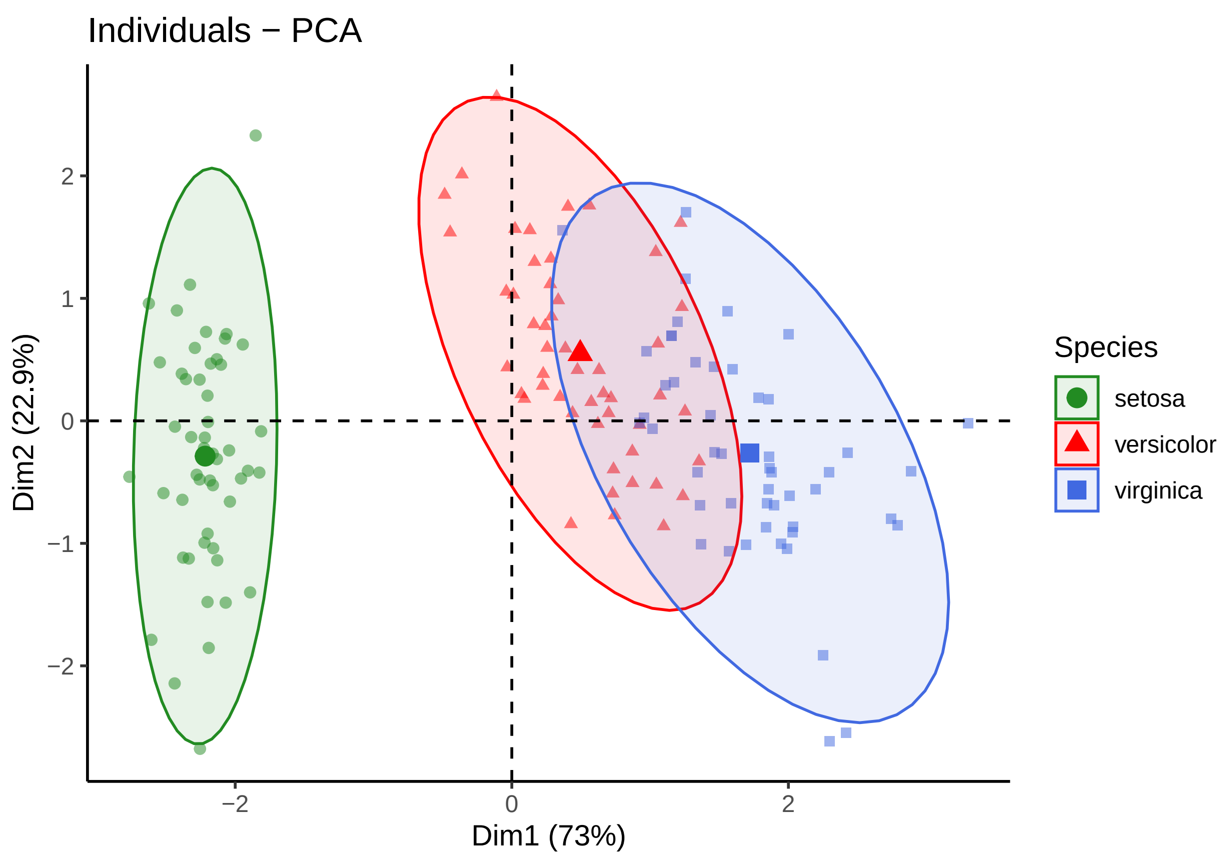
```r
plot.sepal.length = ggplot2::ggplot(data = iris_data, aes(x = species, y=sepal_length,
                                     fill=species)) +
  geom_boxplot(alpha = 0.5) +
  scale_fill_manual(values = c("forestgreen",  "red", "royalblue")) +
  stat_summary(fun = mean, geom = "point", shape = 18, size = 5, fill="black") +
  theme(legend.position="none") +
  xlab("Species") +
  ylab("Sepal length (cm)") +
  theme_classic() +
  theme(legend.position="none")

# SEPAL WIDTH
plot.sepal.width = ggplot2::ggplot(data = iris_data, aes(x = species, y=sepal_width,
                                     fill=species)) +
  geom_boxplot(alpha = 0.5) +
  scale_fill_manual(values = c("forestgreen",  "red", "royalblue")) +
  stat_summary(fun = mean, geom = "point", shape = 18, size = 5, fill="black") +
  theme(legend.position="none") +
  xlab("Species") +
  ylab("Sepal width (cm)") +
  theme_classic() +
  theme(legend.position="none")

# PETAL LENGTH
plot.petal.length = ggplot2::ggplot(data = iris_data, aes(x = species, y=petal_length,
                                     fill=species)) +
  geom_boxplot(alpha = 0.5) +
  scale_fill_manual(values = c("forestgreen",  "red", "royalblue")) +
  stat_summary(fun = mean, geom = "point", shape = 18, size = 5, fill="black") +
  theme(legend.position="none") +
  xlab("Species") +
  ylab("Petal length (cm)") +
  theme_classic() +
  theme(legend.position="none")

# PETAL WIDTH
plot.petal.width = ggplot2::ggplot(data = iris_data, aes(x = species, y=petal_width,
                                     fill=species)) +
  geom_boxplot(alpha = 0.5) +
  scale_fill_manual(values = c("forestgreen",  "red", "royalblue")) +
  stat_summary(fun = mean, geom = "point", shape = 18, size = 5, fill="black") +
  theme(legend.position="none") +
  xlab("Species") +
  ylab("Petal width (cm)") +
  theme_classic() +
  theme(legend.position="none")

# let's combine these all onto one plot
ggpubr::ggarrange(plot.sepal.length, plot.sepal.width,
                  plot.petal.length, plot.petal.width,
          labels = c("A", "B", "C", "D"),
          ncol = 2, nrow = 2)
```



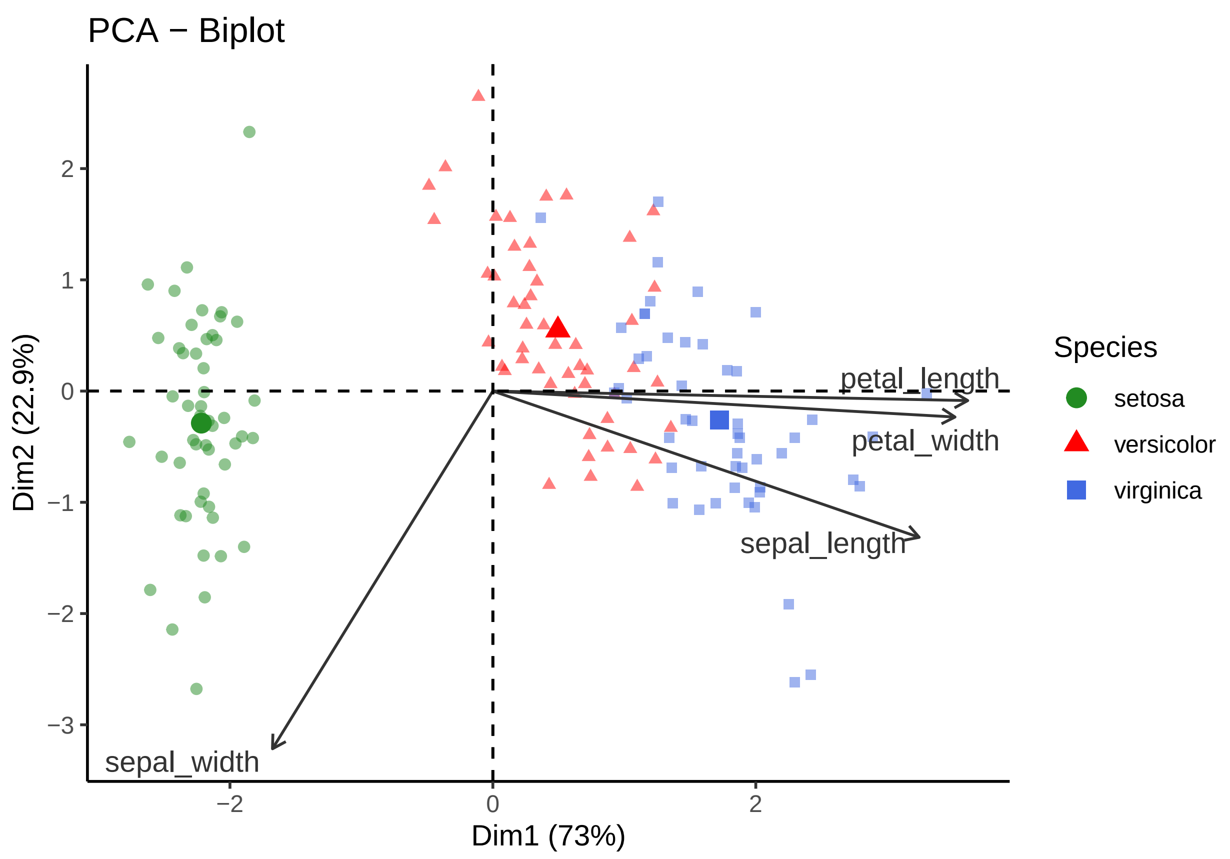
```
# plot the PCA
factoextra::fviz_pca_ind(pca_iris, geom = "point",
                         col.ind = iris_data$species,
                         palette = c("forestgreen",  "red", "royalblue"),
                         legend.title = "Species", alpha.ind = 0.6) +
  scale_shape_manual(values=c(19, 17, 15)) +
  theme_classic()
```



```r
# Have a look at what it looks like when we add ellipses around groups
factoextra::fviz_pca_ind(pca_iris, geom = "point",
                         col.ind = iris_data$species,
                         palette = c("forestgreen",  "red", "royalblue"),
                         legend.title = "Species", alpha.ind = 0.5,
                         addEllipses = TRUE) +
  scale_shape_manual(values=c(19,17,15)) +
  theme_classic()
```



```r
# a biplot
factoextra::fviz_pca_biplot(pca_iris, repel = TRUE,
                            geom = "point",
                            col.ind = iris_data$species,
                            palette = c("forestgreen",  "red", "royalblue"),
                            alpha.ind = 0.5,
                            legend.title = "Species",
                            col.var = "grey20" ) +
  scale_shape_manual(values=c(19,17,15)) +
  theme_classic()
```



Plot a 3D PCA for fun!

```r
# Define colors for each species
species_cols <- c("forestgreen",  "red", "royalblue")  # You can choose any colors you like

# Assign colors to each data point based on species
colors <- species_cols[iris_data$species]

rgl::plot3d(pca_iris$x[,1:3], col = colors, type = "s", size = 1)
```