PCA Tutorial

Heath Blackmon

4/12/2018

Contents

Packages and data

Today we will use two new packages

```
install.packages("car")
install.packages("FactoMineR")
```

1) Load the iris data and take a look at what you have

```
data('iris')
head(iris)
```

```
##   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
```

Basic PCA

2) Perform the PCA using the base function `prcomp` use the assignment operator `<-` to send this to a new variable named `pca`.

```
pca <- prcomp(iris[, 1:4])
```

3) Review the object that is returned. You can see the names for the different elements in the list using the `names` function (`names(pca)`).

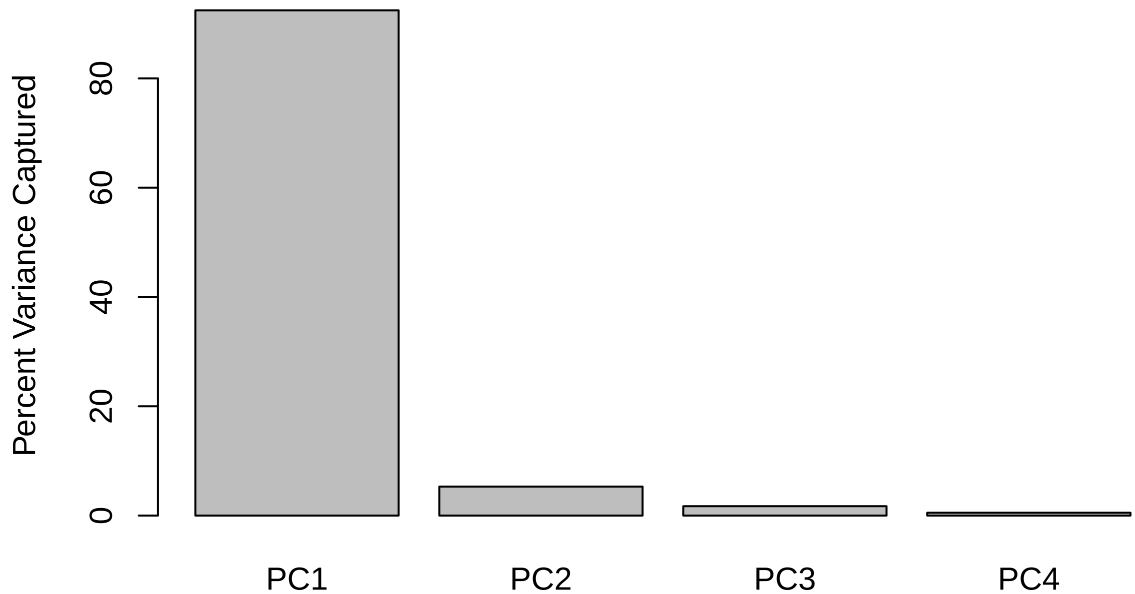
```
names(pca)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

4) A scree plot shows us the proportion of variance explained by each principal component. Use the first element `pca$sdev` along with the `sum` function and square function (`^2`) to create a scree plot.

```
y <- pca$sdev^2 / sum(pca$sdev^2) * 100
barplot(y,
        names.arg=c("PC1", "PC2", "PC3", "PC4"),
```

```r
        ylab = "Percent Variance Captured",
        xlab = "",
        main = "")
```

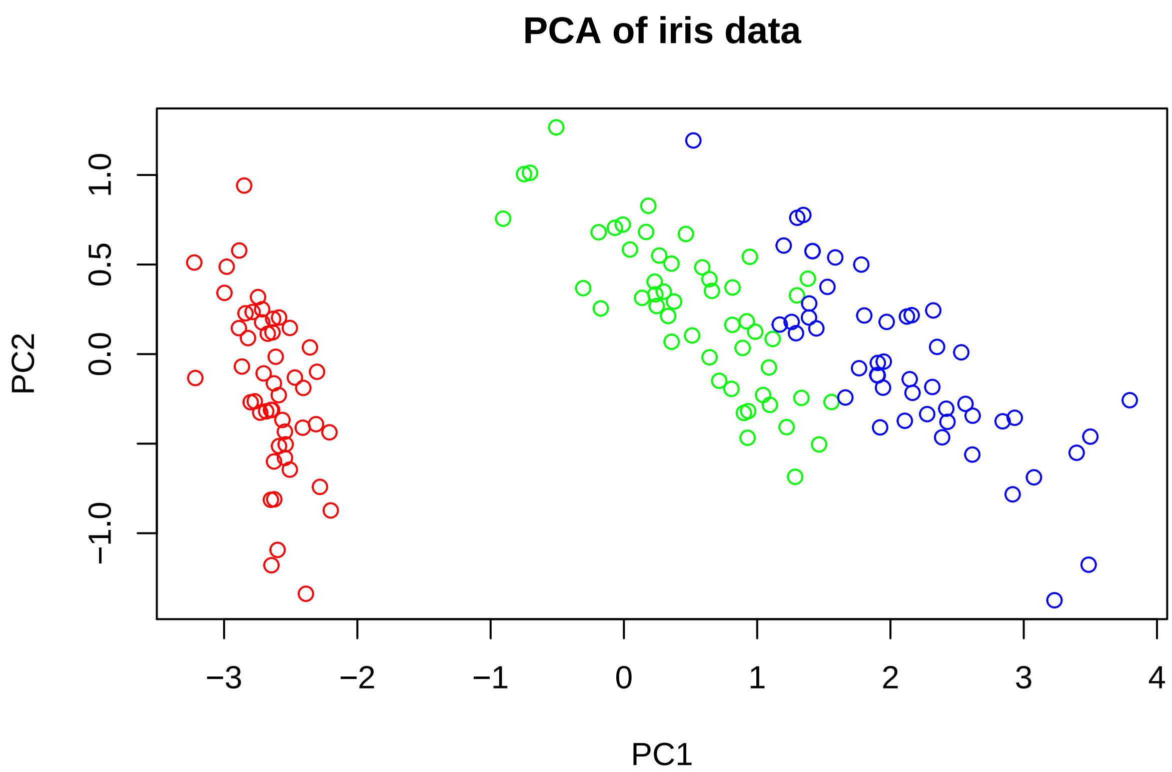


Results of PCA

5) Use the fifth element (`pca$x`) to create a plot of the samples in principal component space. Color each point based on its species identity.

-hint: factors have values of 1, 2, 3. So what would you get if you ran this code

```r
plot(x = pca$x[, 1],
     y = pca$x[, 2],
     col = rainbow(3)[iris$Species],
     xlab = "PC1",
     ylab = "PC2",
     main = "PCA of iris data")
```



Loadings and variables factor map

The second element (`pca$rotation`) has loadings – this is the correlation between each of your raw measures and the sample scores for each principal component. Sometimes people will report these raw correaltions.

```
pca$rotation
```

```
##                     PC1         PC2         PC3        PC4
## Sepal.Length  0.36138659 -0.65658877  0.58202985  0.3154872
## Sepal.Width  -0.08452251 -0.73016143 -0.59791083 -0.3197231
## Petal.Length  0.85667061  0.17337266 -0.07623608 -0.4798390
## Petal.Width   0.35828920  0.07548102 -0.54583143  0.7536574
```

However, I think a more intuitive measure is how much of the variance in the raw measure is captured by a principal component to get this simply square these values.

```
pca$rotation^2
```

```
##                    PC1         PC2         PC3        PC4
## Sepal.Length 0.130600269 0.431108815 0.338758748 0.09953217
## Sepal.Width  0.007144055 0.533135721 0.357497361 0.10222286
## Petal.Length 0.733884527 0.030058080 0.005811939 0.23024545
## Petal.Width  0.128371149 0.005697384 0.297931952 0.56799951
```

6) Which raw variable is explained the most by PC1 what about PC2

```
row.names(pca$rotation)[which.max(pca$rotation[,1]^2)]
```

```
## [1] "Petal.Length"
```

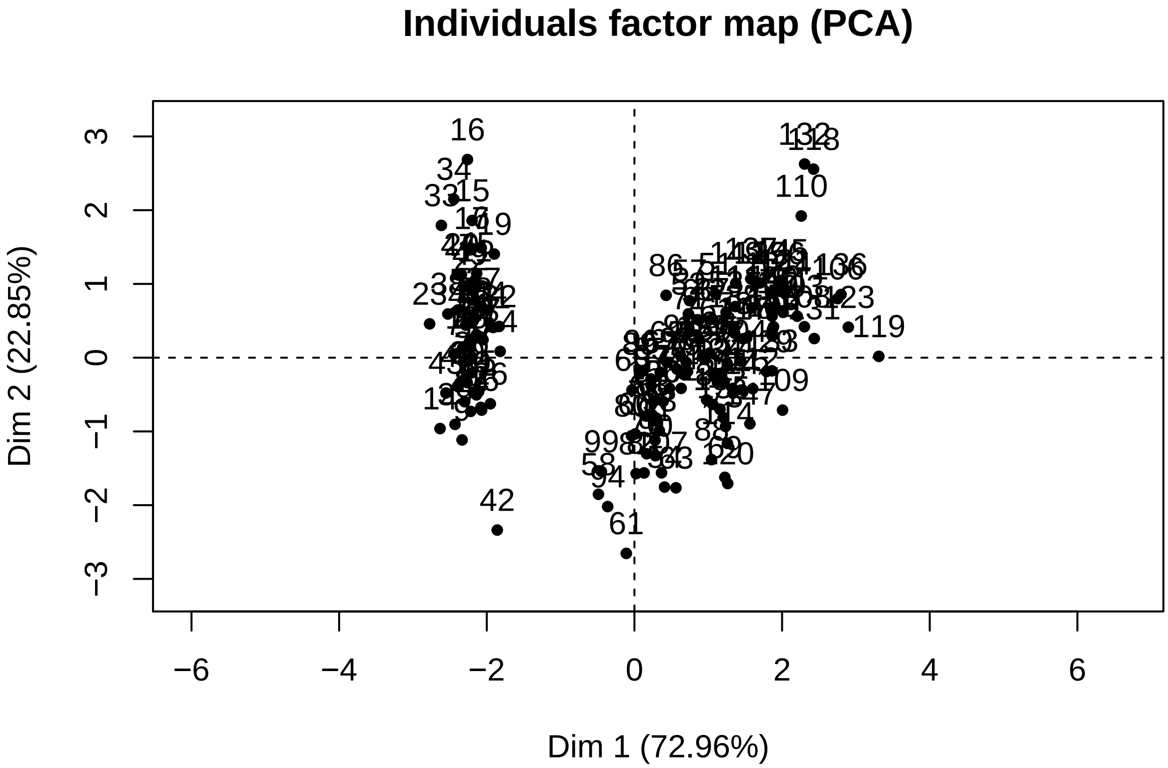
Rather than reporting these values sometimes you will see a plot called a variables factor map. It is a graphical representation of the loadings. The package `FactoMineR` offers a nice way to produce this plot.

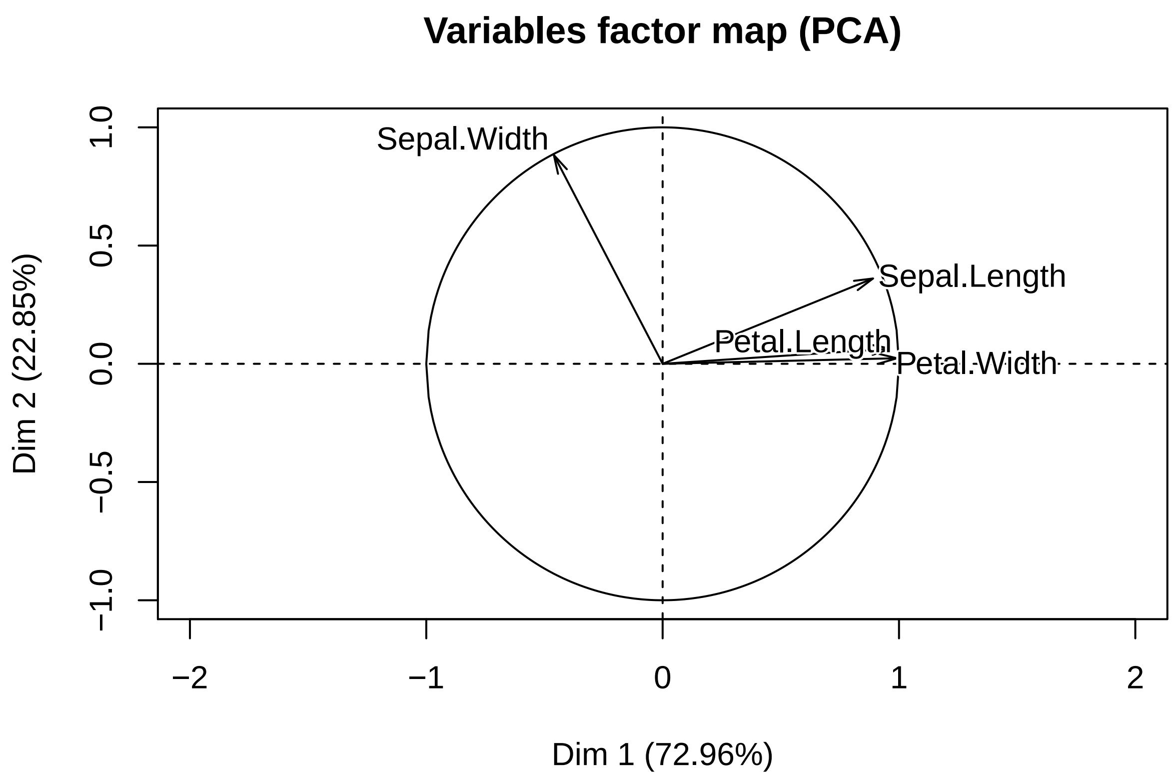
7) Repeat your PCA using this package. In this package the pca is done with the function `PCA` and you can set `graph = TRUE` to autimatically produce the variables factor map.

```
library(FactoMineR)
```

```
## Warning: package 'FactoMineR' was built under R version 3.4.4
```

```
pca3 <- PCA(iris[,1:4], graph = T)
```





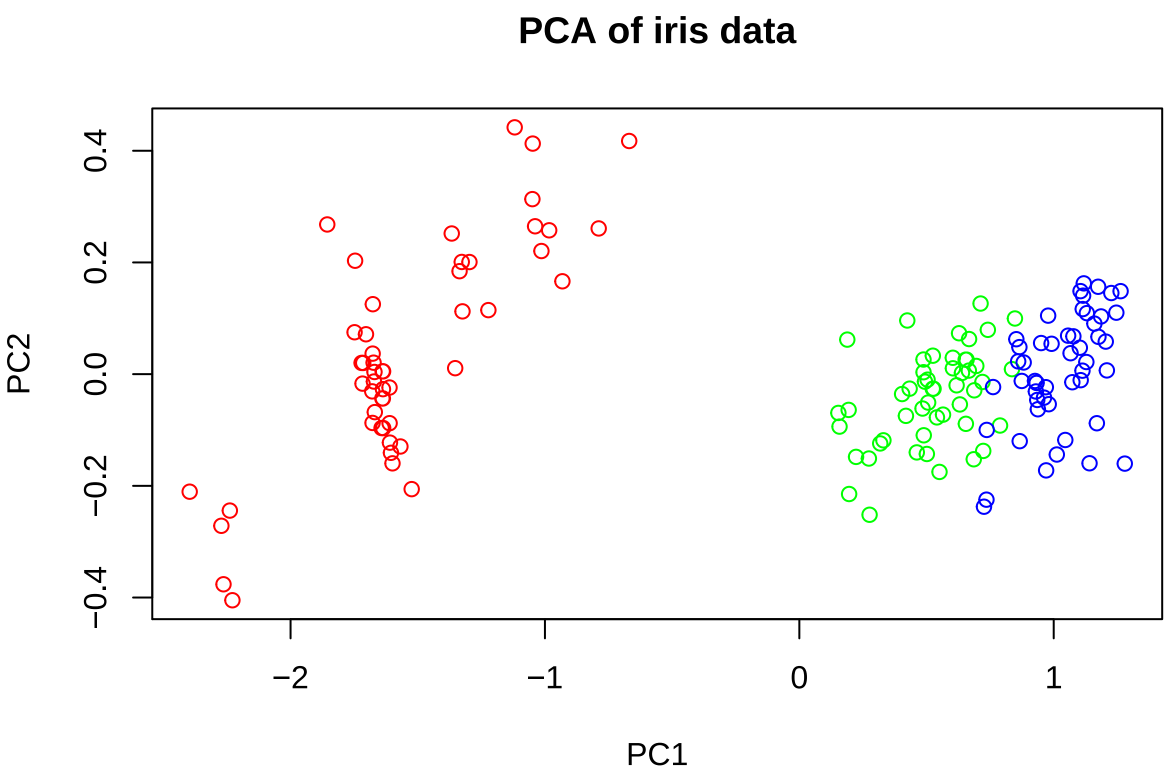
Input data and assumptions

The third and fourth elements (`pca$center` and `pca$scale`) of the list returned by `prcomp` are center and scale these items describe the transformation that is performed on the data before analysis.

PCA does make some assumptions about the input data the most important of these is that measured variables either have linear or no correlation. In practice, this is often untested. Instead, researchers will often log transform all of their data before doing the PCA.

8) Try log transforming the iris data does this change the result?

```
iris$Sepal.Length <- log(iris$Sepal.Length)
iris$Sepal.Width <- log(iris$Sepal.Width)
iris$Petal.Length <- log(iris$Petal.Length)
iris$Petal.Width <- log(iris$Petal.Width)
pca2 <- prcomp(iris[, 1:4])
plot(x = pca2$x[, 1],
     y = pca2$x[, 2],
     col = rainbow(3)[iris$Species],
     xlab = "PC1",
     ylab = "PC2",
     main = "PCA of iris data")
```

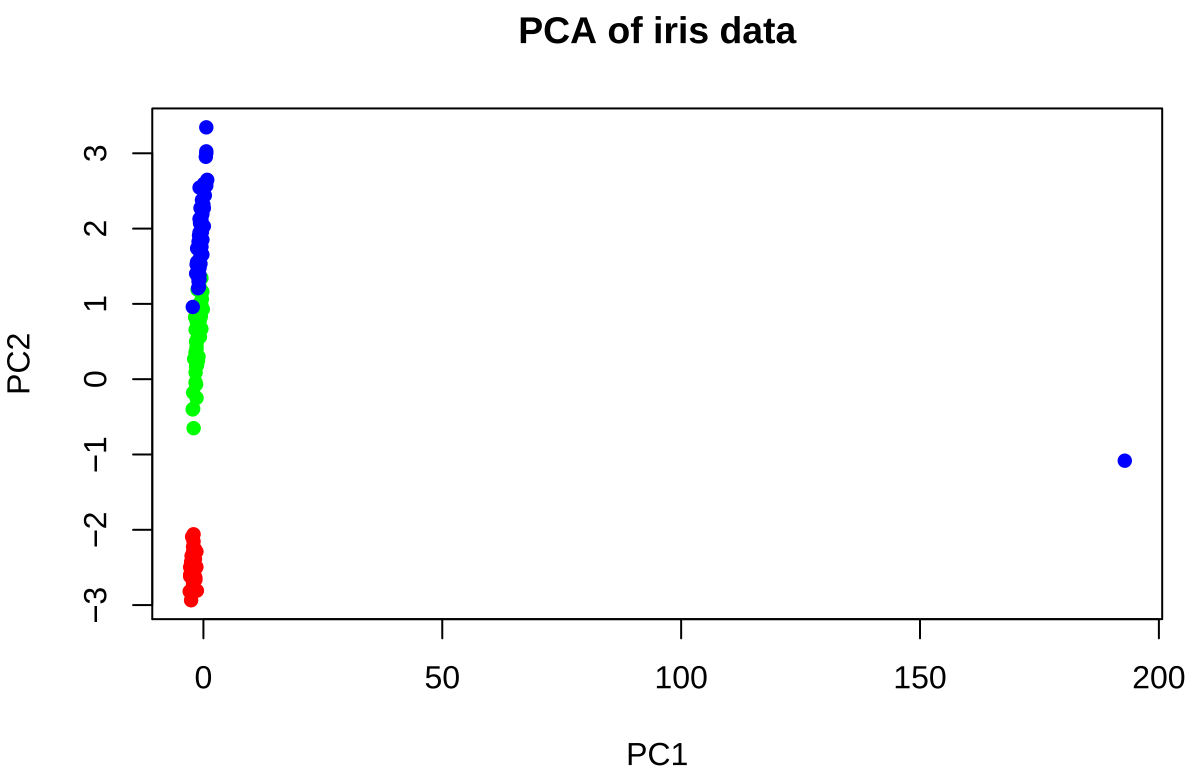


9) PCA is quite sensitive to outliers. Lets alter one data measurement to illustrate this:

```
data(iris)
iris[150, 1] <- 200
```

10) Now perform the PCA again on this dataset and look at the result by plotting the data in the dimensions of the first two principal components.

```
pca3 <- prcomp(iris[, 1:4])
plot(x = pca3$x[, 1],
     y = pca3$x[, 2],
     pch = 16,
     col = rainbow(3)[iris$Species],
     xlab = "PC1",
     ylab = "PC2",
     main = "PCA of iris data")
```



Clustering data

Often we would like to understand how well datapoints cluster in principal component space. This is especially useful when we want to determine whether a new data point falls within our existing categories. To do this we will use a function `dataEllipse` from the `car` package. Lets add an unknown specimen to our dataset and try and determine which species we believe it belongs to.

11) follow this code to get a clean copy of your data and add a new datapoint from an unknown species.

```
# lets clear our memory and start fresh
rm(list=ls())

data(iris)
# first we need to convert the species names from
# factors to text
Species <- as.character(iris$Species)

# now we can add our new data
iris[151, 1:4] <- c(7, 3.1, 4.5, 1.3)

# and now we add the new set of species names as factors
iris$Species <- as.factor(c(Species, "unknown"))
```

12) Now repeat your PCA with this data. Plot the result of the PCA and add ellipses for each species. Below I illustrate the basic plot and one ellipse.

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.4.4
```

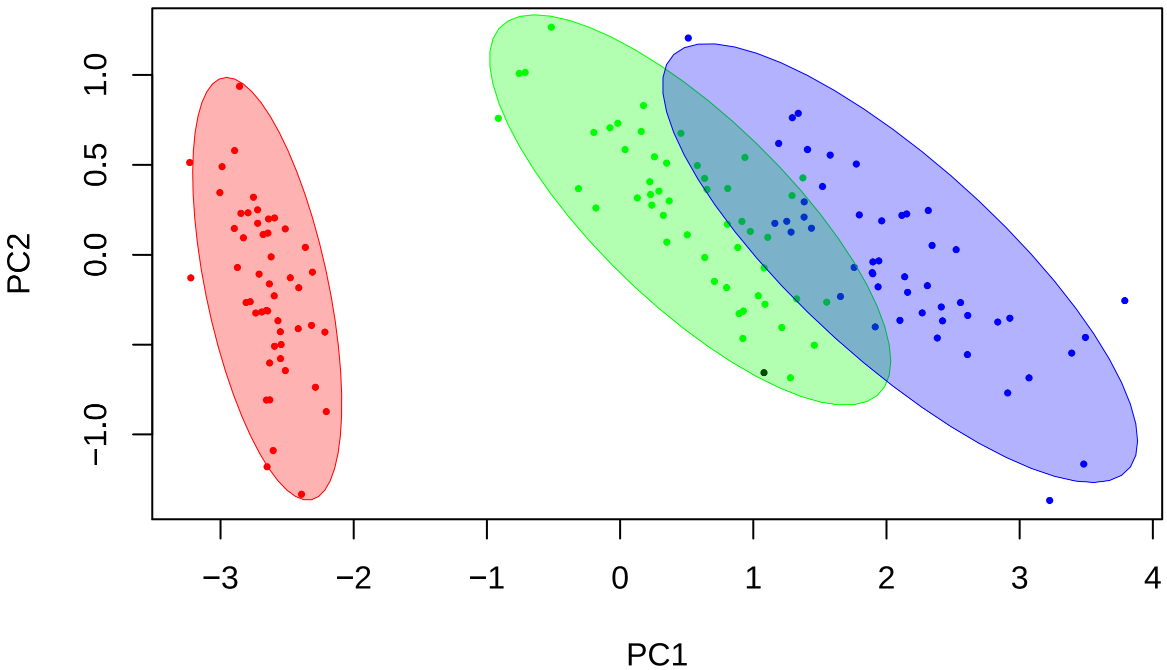
```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 3.4.4
```

```
pca <- prcomp(iris[,1:4])

# examine first PCs
plot(x = pca$x[, 1],
     y = pca$x[, 2],
     col = c("red","black","green", "blue")[iris$Species],
     pch=16,
     cex=.5,
     xlab = "PC1",
     ylab = "PC2")

dataEllipse(x = pca$x[1:50, 1],
            y = pca$x[1:50, 2],
            add = T,
            plot.points = F,
            levels = .95,
            center.pch = F,
            col = "red",
            fill = T,
            lwd=.5)
dataEllipse(x = pca$x[51:100, 1],
            y = pca$x[51:100, 2],
            add = T,
            plot.points = F,
            levels = .95,
            center.pch = F,
            col = "green",
            fill = T,
            lwd=.5)
dataEllipse(x = pca$x[101:150, 1],
            y = pca$x[101:150, 2],
            add = T,
            plot.points = F,
            levels = .95,
            center.pch = F,
            col = "blue",
            fill = T,
            lwd=.5)
```


PC2
1.0
0.5
0.0
−1.0
−3
−2
−1
0
1
2
3
4
PC1