```
log.ir <- log(iris[, 1:4])
```

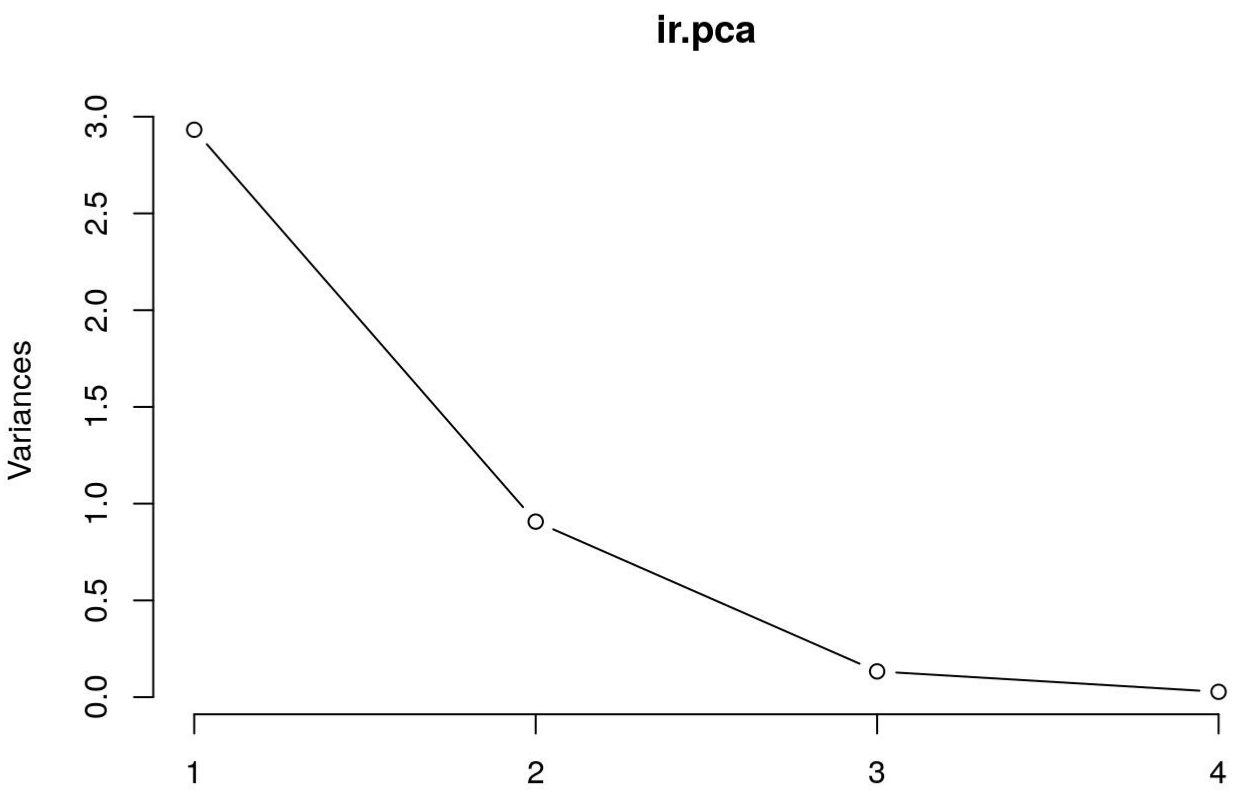
Log transformation for scaling the data
Because its good to transform the skewness by its magnitude

```
ir.pca <- prcomp(log.ir, center = TRUE, scale. = TRUE)
```

Prcomp -The print method returns the standard deviation of each of the four PCs, and their rotation (or loadings), which are the coefficients of the linear combinations of the continuous variables.
And stored it into ir.pca

Rotation(n*k)= 4*4
Pc1,pc2,pc3,pc4

```
plot(ir.pca, type = "l")
```



The plot method returns a plot of the variances (y-axis) associated with the PCs (x-axis).

First two pcs explain most -of the variability in the data

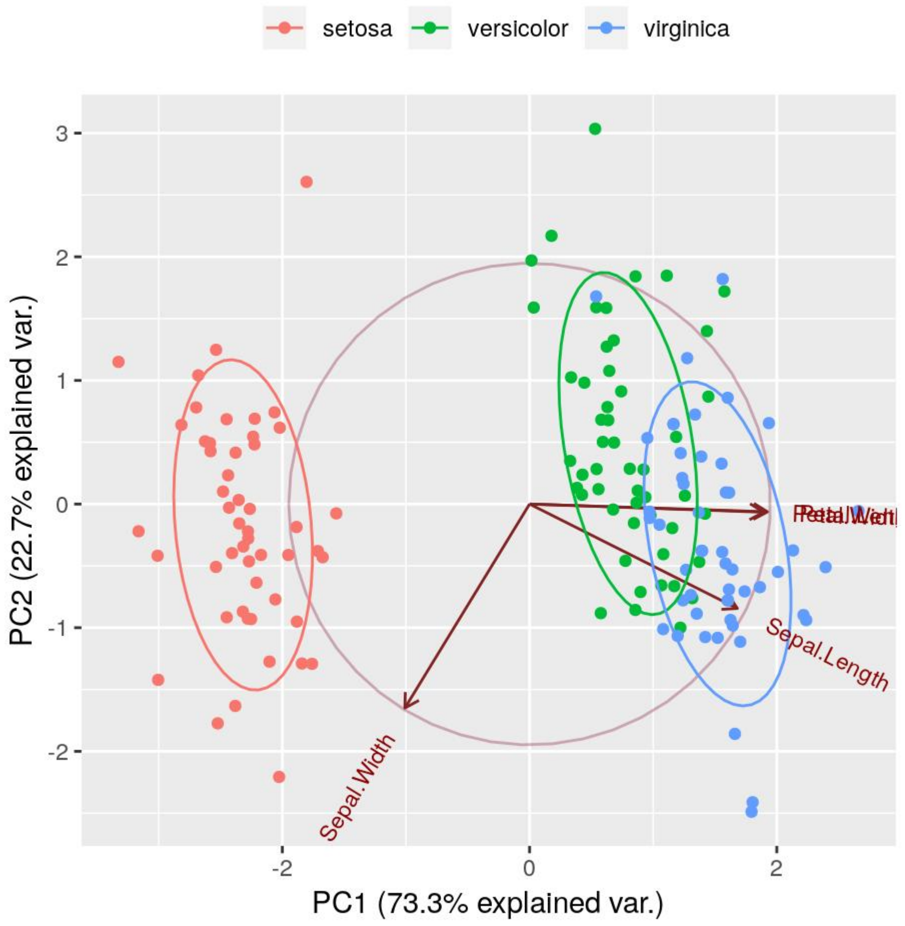
```
summary(ir.pca)
```

The summary method describe the importance of the PCs. The first row describe again the standard deviation associated with each PC. The second row shows the proportion of the variance in the data explained by each component while the third row describe the cumulative proportion of explained variance. We can see there that the first two PCs accounts for more than {95%} of the variance of the data.

After liberay
Devtools
Ggbiplot

```
g <- ggbiplot(ir.pca, obs.scale = 1, var.scale = 1,

              groups = ir.species, ellipse = TRUE,

              circle = TRUE)

g <- g + scale_color_discrete(name = '')

g <- g + theme(legend.direction = 'horizontal',

               legend.position = 'top')

print(g)
```

I also like to plot each variables coefficients inside a unit circle to get insight on a possible interpretation for PCs.



```
ggplot(df, aes(Petal.Length, Petal.Width)) + geom_point(aes(col=Species), size=4
```

Scattering the plot of species length vs petals width

```
set.seed(101)
irisCluster <- kmeans(df[,1:4], center=3, nstart=20)
irisCluster
```

K-means clustering with 3 clusters into dataframe of 1 ot 4

```
table(irisCluster$cluster, df$Species)
```

Clusters matrix of species

```
clusplot(iris, irisCluster$cluster, color=T, shade=T, labels=0, lines=0)
```

Plotting the variabilty of first two component is 95.02%

```
tot.withinss <- vector(mode="character", length=10)
for (i in 1:10){
  irisCluster <- kmeans(df[,1:4], center=i, nstart=20)
  tot.withinss[i] <- irisCluster$tot.withinss
}
```

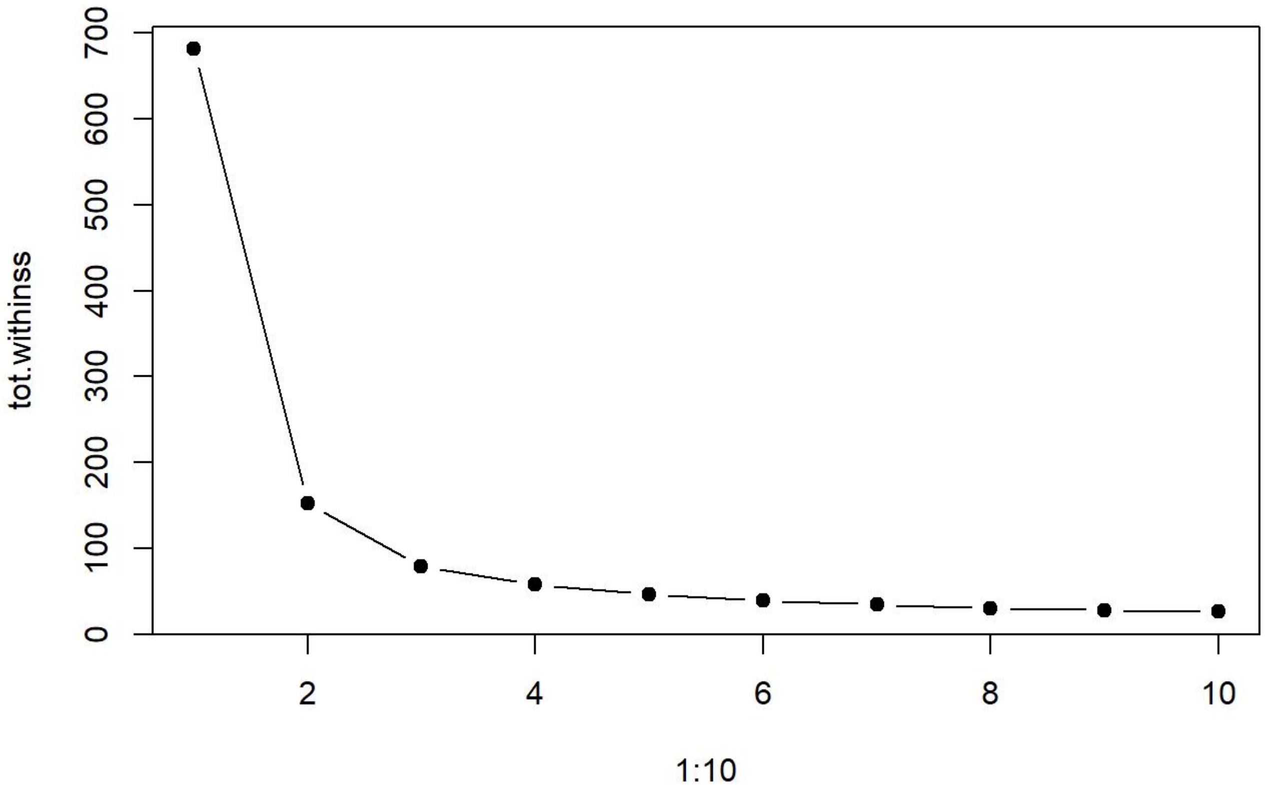
We can see the setosa cluster perfectly explained, meanwhile virginica and versicolor have a little noise between their clusters.

As I said before, we will not always have the labeled data. If we would want to know the exactly number of centers, we should have built the elbow method.

```
tot.withinss <- vector(mode="character", length=10)for (i in 1:10){

  irisCluster <- kmeans(df[,1:4], center=i, nstart=20)

  tot.withinss[i] <- irisCluster$tot.withinss

}
```

Let’s visualize it.

```
plot(1:10, tot.withinss, type="b", pch=19)
```



As we saw, the optimal number of clusters is 3.