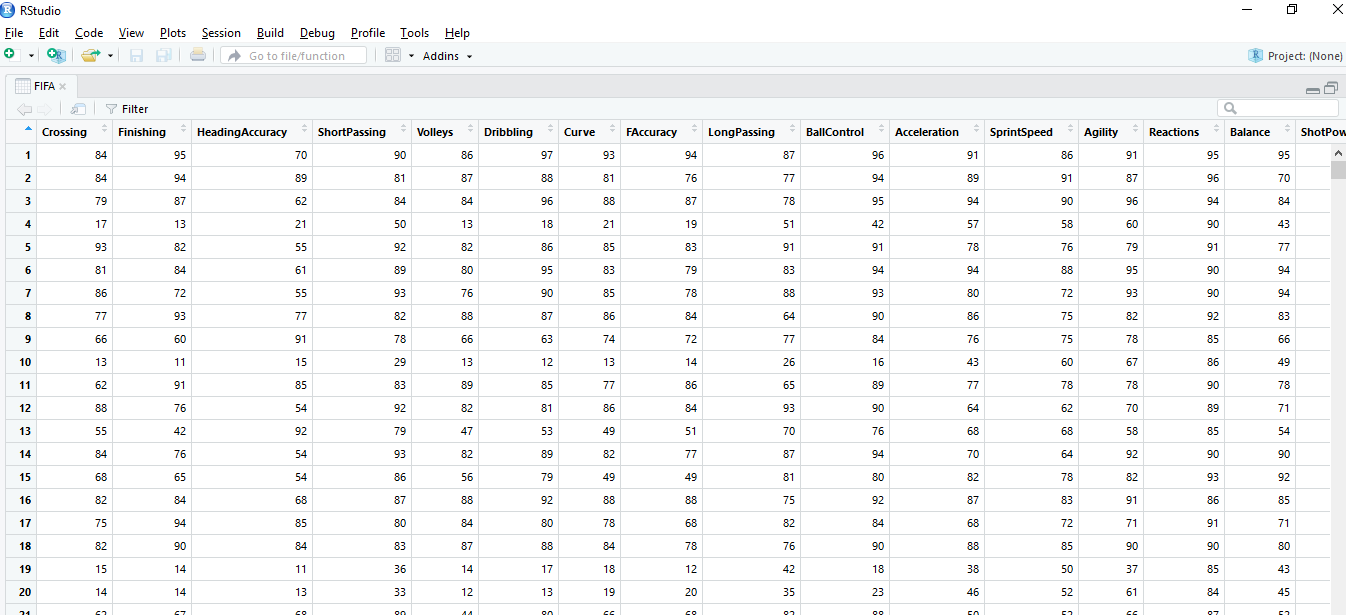
**PRINCIPLE COMPONENT ANALYSIS**

PCA is a statistical technique used to reduce the dimension of the dataset. This analysis is carried out on a FIFA data set consisting of 34 variables like name, age, their game career scores, strength, crossing etc and 12784 observations. The idea behind PCA is to extract important variables that provide maximum information.

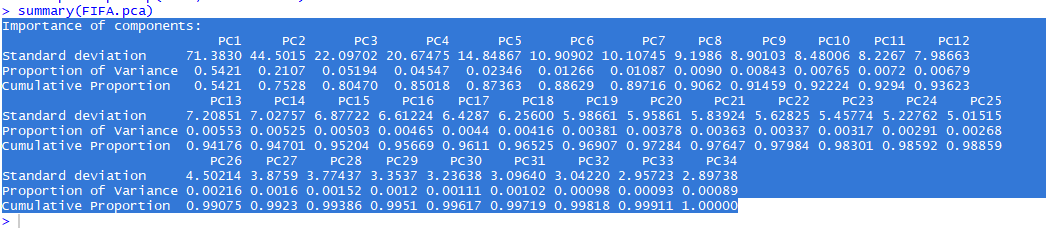
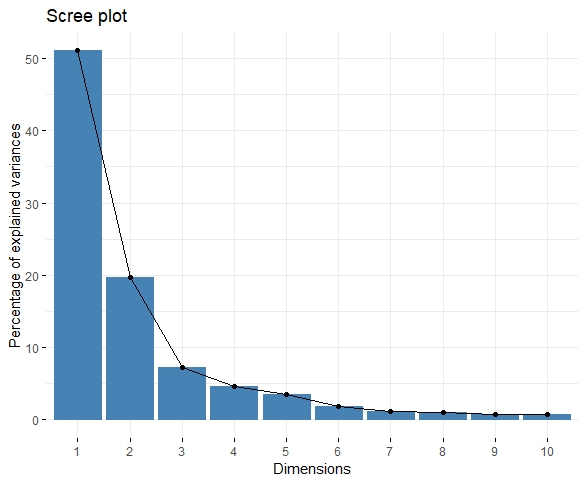
**View(FIFA)**

****

**Running PCA analysis on the above dataset:**

**FIFA.pca = prcomp(FIFA,center= TRUE)**

**Summary(FIFA.pca)**

PCA assumes that the directions with the largest variances are the most “important” (i.e, the most principal).From the above summary statistics , we can see that PC1 explains 51.15% of the variation in the data and PC2 explains 19.8% and so. 90% of the variance is explained by PC1, PC2, PC3, PC4, PC5, PC6 and PC7. Thus 34 dimensional data can be reduced to 7 components that explain maximum variance. **** 

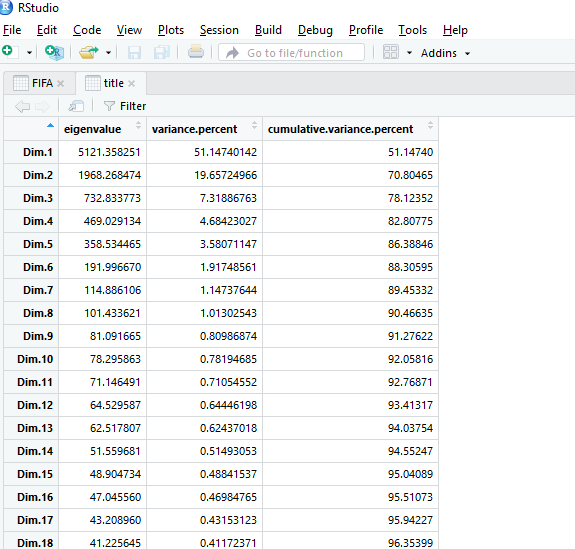
**The above graph between percentage of variance and dimension puts the above stats to perspective.**

**fviz\_eig(FIFA.pca)**

**Eigen Values and vectors for the PCA analysis:**

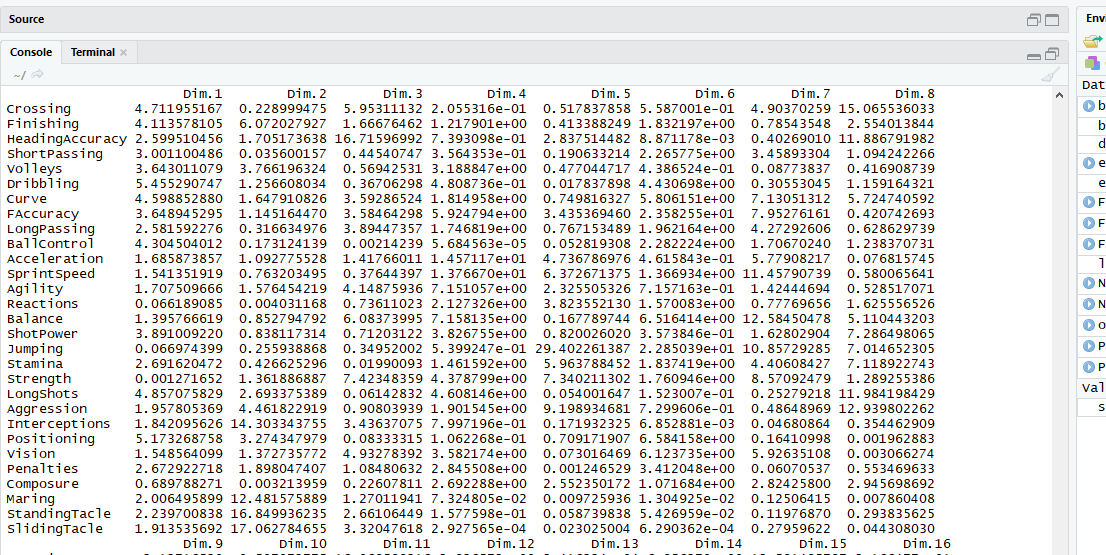
**eig.val = get\_eigenvalue(FIFA.pca)**

**> view(eig.val)**

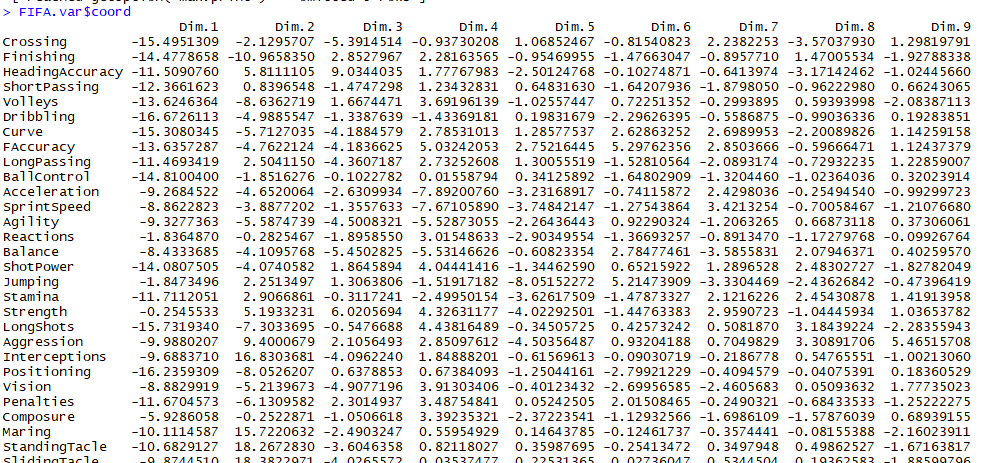


**Contribution of each variable to the principle components:**

**FIFA.var$contrib**

****

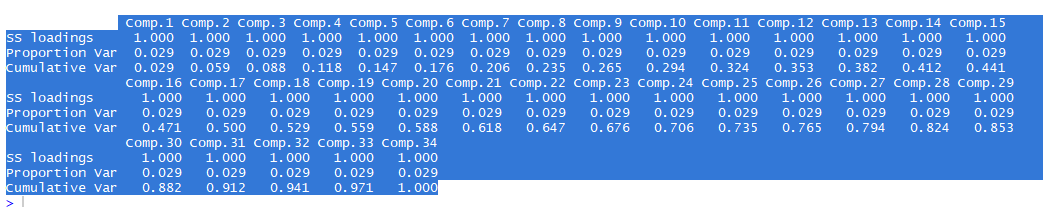
**FIFA.var$coord**

****

**The loadings () function were expecting the eigenvectors to be normalized to the corresponding eigenvalue. The first line reported by loadings, "SS loadings" has its columns normalized to 1.**

**othercomp<-princomp(FIFA)**

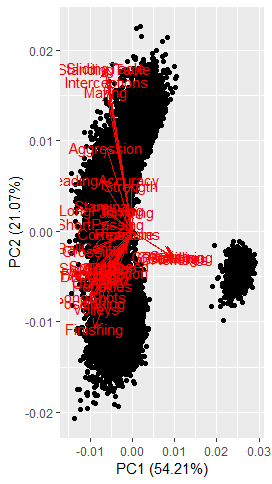
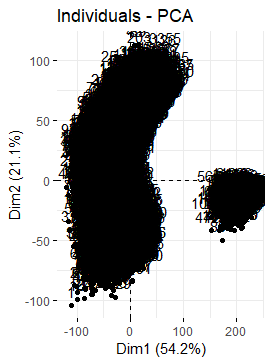
**summary(othercomp)**

****

**Below is the graph between PC1 and PC2 and background plot of all the variables**

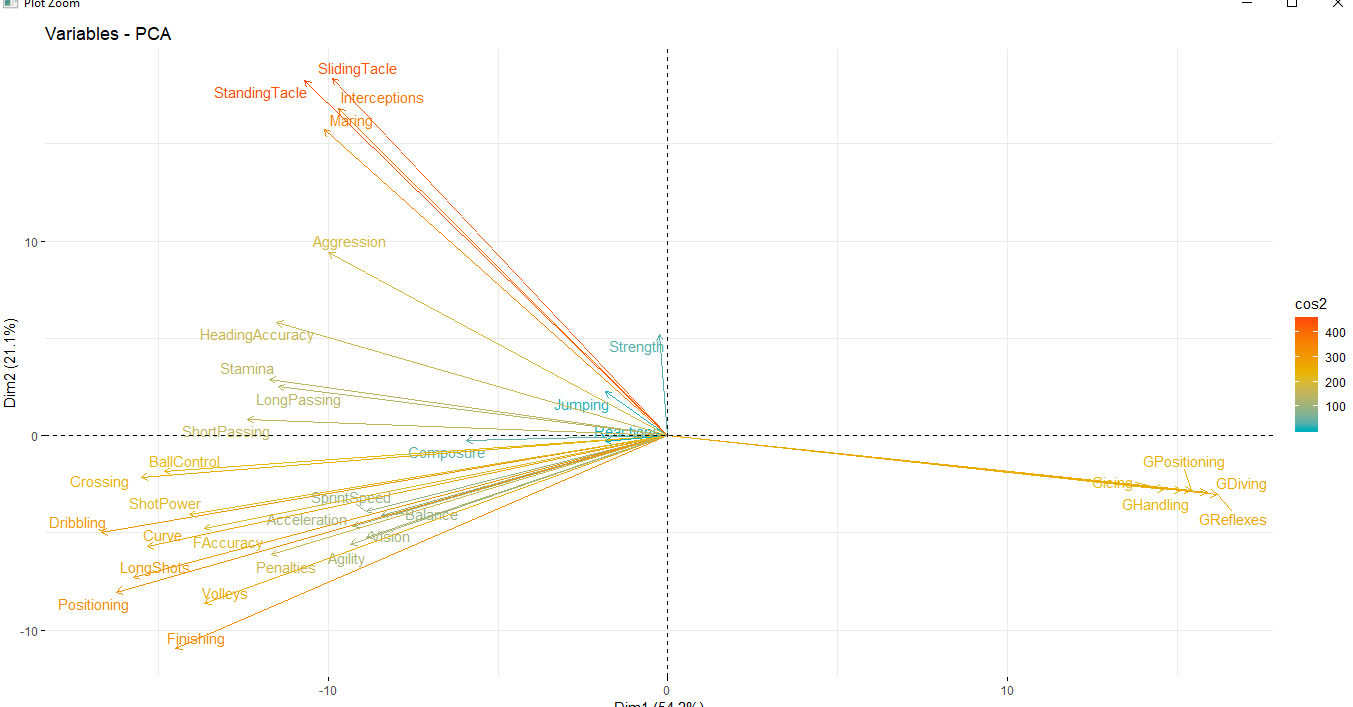
**fviz\_pca\_ind(FIFA.pca)**

**autoplot(FIFA.pca,loadings= TRUE,loadings.label= TRUE)**



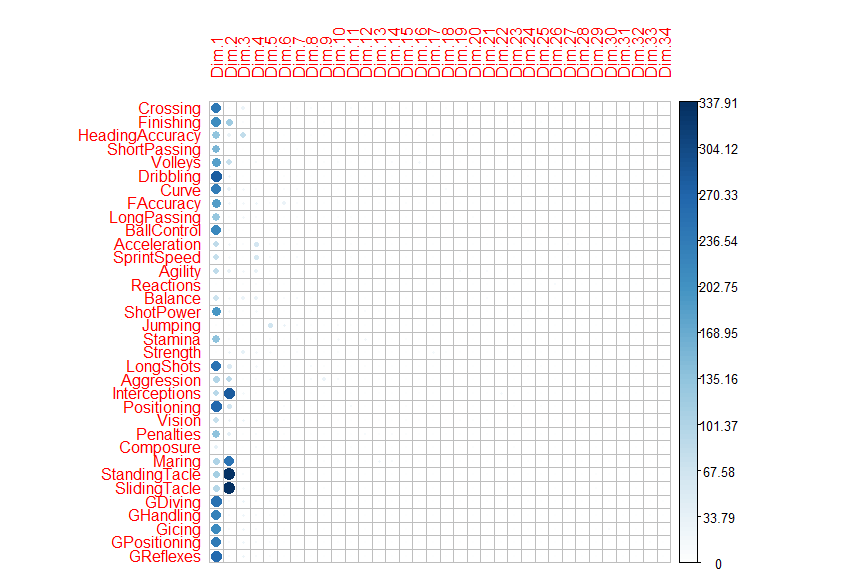
**Different way of visualizing dimensions and variables:**

fviz\_pca\_var(FIFA.pca, col.var = "cos2",+ gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),+ repel = TRUE # Avoid text overlapping

****

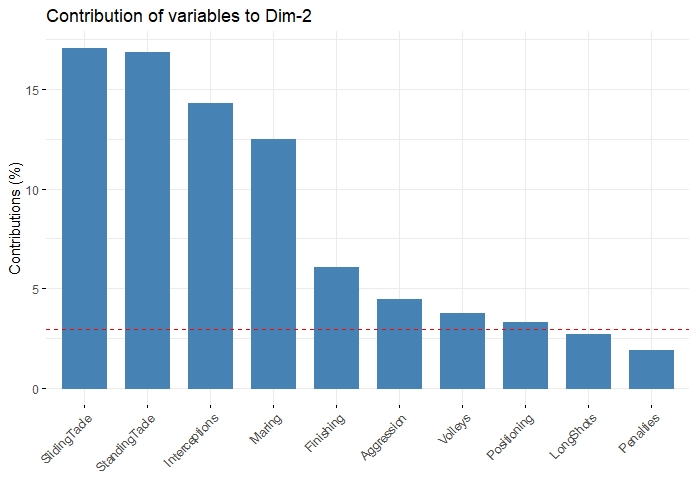
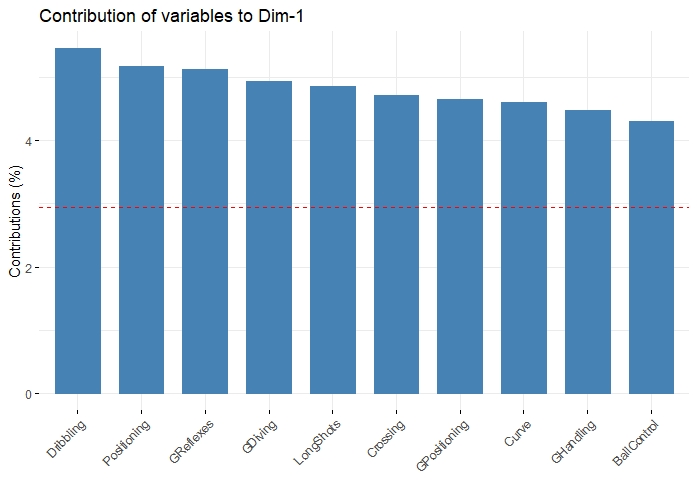
**Visualizing the cos2 of the variables on all the dimensions:**

**corrplot(FIFA.var$cos2, is.corr=FALSE)**

****

**The total contribution of variables to PC1 and Pc2 are obtained as below:**

**fviz\_contrib(FIFA.pca, choice = "var", axes = 1:2, top = 10)**

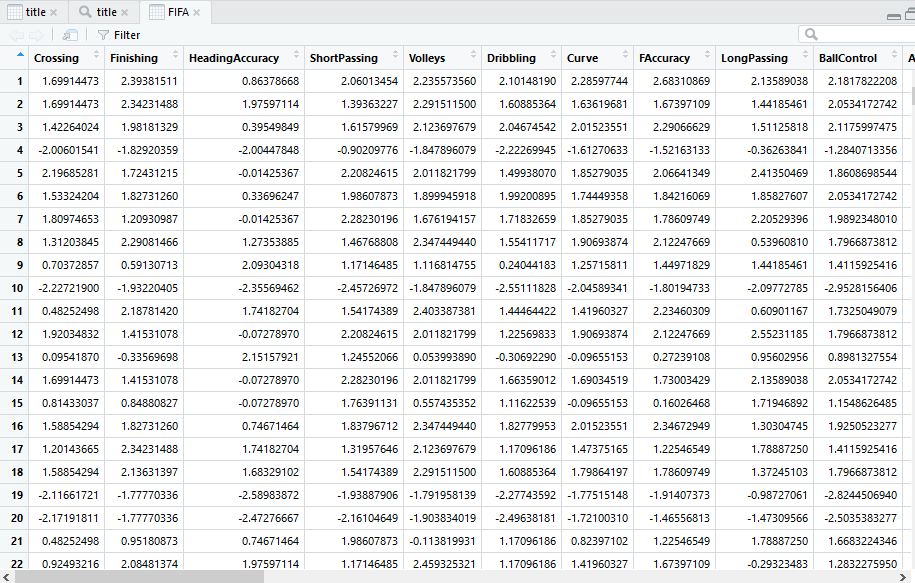
****

**CLUSTER ANALYSIS**

R is a great tool for cluster analysis. Cluster analysis is an unsupervised learning method of grouping similar data together such that objects in one group are significantly different from the objects in other groups.

Before we begin cluster analysis, let us scale the data

FIFA<-scale(FIFA)



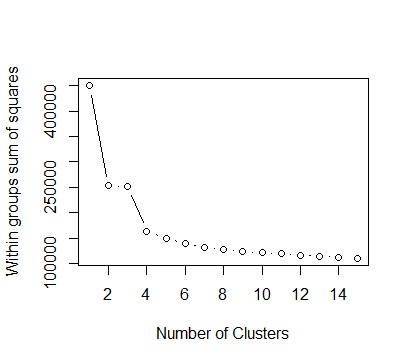
**K MEANS CLUSTERING**

K-means is the most popular approach for clustering data.

For the k means approach, we have to specify the number of clusters. It will be helpful to plot “within groups sum of squares” with “number of clusters” to determine number of clusters.

**wss <- (nrow(FIFA)-1)\*sum(apply(FIFA,2,var)) > for (i in 2:15) wss[i] <- sum(kmeans(FIFA,**

**+ centers=i)$withinss) > plot(1:15, wss, type="b", xlab="Number of Clusters", + ylab="Within groups sum of squares")**



From the above plot it is evident that after 4 clusters the observed difference in within cluster dissimilarity is insignificant. Hence choosing 4 clusters is a good place to start at.

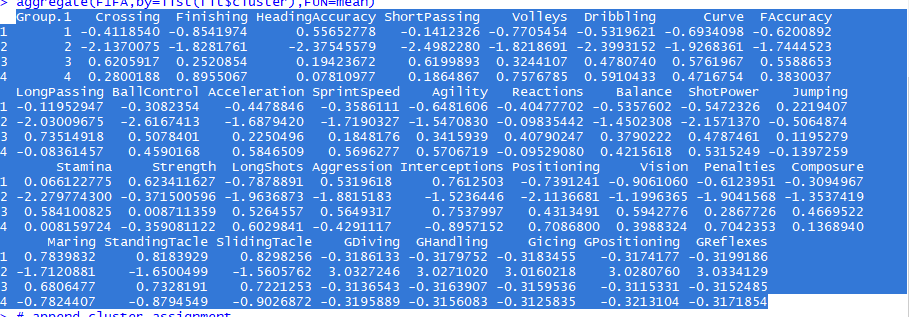
**Fitting the data to the k means cluster :**

**fit <- kmeans(FIFA, 4) # 5 cluster solution >**

**> aggregate(FIFA,by=list(fit$cluster),FUN=mean)**

**mydata <- data.frame(FIFA, fit$cluster)**

By running the above, we can get distance of each variable from the mean of all the 4 clusters.



**Plotting clusters:**

**library(cluster)**

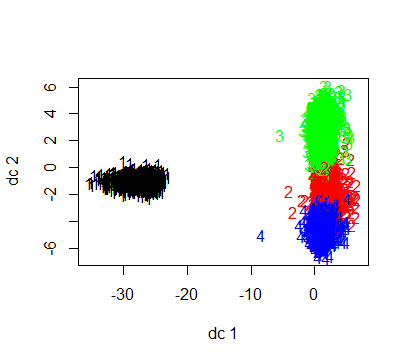
**> library(fpc)**

**> dat <- mydata[, ]**

**> # Kmeans clustre analysis**

**> clus <- kmeans(dat, centers=4)**

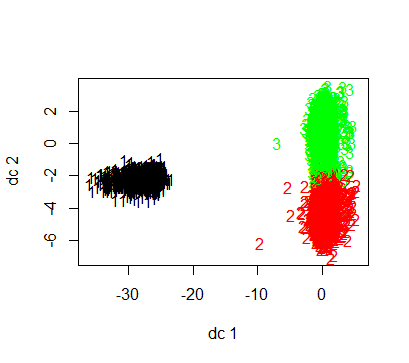
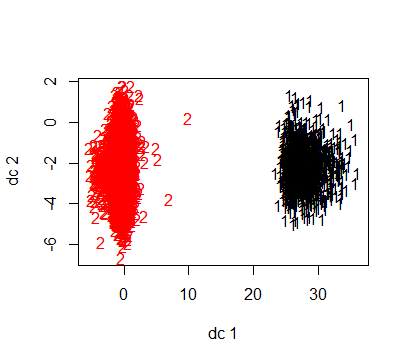
**> plotcluster(dat, clus$cluster)**



The above plot clearly highlights that there are 3 overlapping clusters. We can choose to have 2 or 3 distinct clusters for this dataset rather than 4 clusters.

Let’s replot with 2 and 3 clusters.

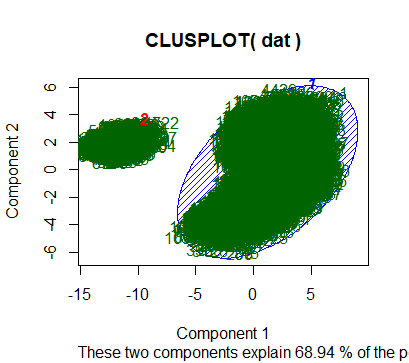
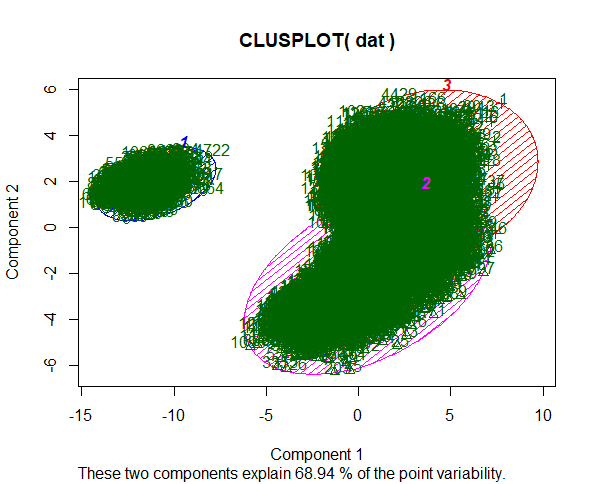
**Thus we can decide on 2 clusters for this dataset.**



**Another representative way to plot clusters using clustplot:**

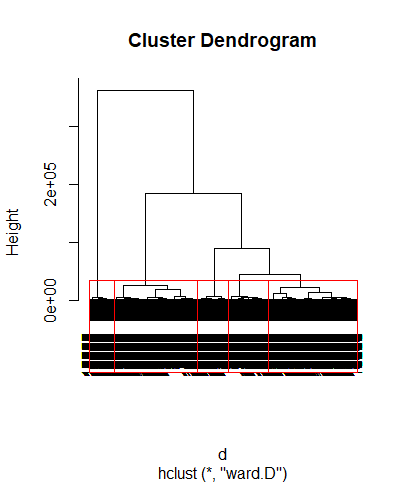
**clusplot(dat, clus$cluster, color=TRUE, shade=TRUE, labels=2, lines=0)**

**clusplot(dat, clus$cluster, color=TRUE, shade=TRUE, labels=3, lines=0)**

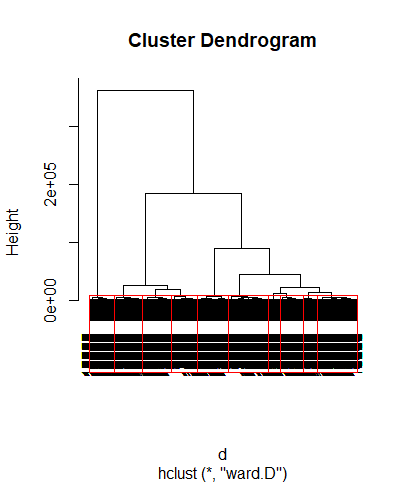
**Hierarchical Agglomerative:**

d <- dist(mydata, method = "euclidean")   
fit <- hclust(d, method="ward")   
plot(fit) # display dendogram  
groups <- cutree(fit, k=5)   
# draw dendogram with red borders around the 5 clusters   
rect.hclust(fit, k=5, border="red")



For 10 clusters:

d <- dist(mydata, method = "euclidean")   
fit <- hclust(d, method="ward")   
plot(fit) # display dendogram  
groups <- cutree(fit, k=10)   
# draw dendogram with red borders around the 5 clusters   
rect.hclust(fit, k=10, border="red")



|  |
| --- |
|  |
| |  | | --- | |  | |