Assignment 5

Sonali Singh

knitr::opts\_chunk$set(echo = TRUE)

# Unsupervised Learning

Unsupervised learning is a class of machine learning algorithms to identify patterns or grouping structure in the data. Unlike supervised learning which relies on “supervised” information such as the dependent variable to guide modeling, unsupervised learning seeks to explore the structure and possible groupings of unlabeled data. This information will be useful to provide pre-processor for supervised learning.

Unsupervised learning has no explicit dependent variable of Y for prediction. Instead, the goal is to discover interesting patterns about the measurements on and identify any subgroups among the observations.

Generally, in this section, the two general methods are introduced: Principal components analysis and Clustering.

## Principal Component Analysis (PCA)

Principal Components Analysis (PCA) produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.

The first principal component of a set of features is the normalized linear combination of the features:

that has the largest variance. By normalized, we mean that .

The elements are the loadings of the first principal component; together, the loadings make up the principal component loading vector,

We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

## Clustering

### K-Means Clustering

The K-means clustering method is to partition the data points into k groups such that the sum of squares from points to the assigned cluster center in each group is minimized.

### Hierarchical Clustering

Hierarchical clustering is an alternative approach which does not require a pre-specified or a particular choice of .

Hierarchical Clustering has an advantage that it produces a tree-based representation of the observations: Dendrogram

A dendrogram is built starting from the leaves and combining clusters up to the trunk. The result of hierarchical clustering is a tree-based representation of the objects, which is also known as dendrogram. Observations can be subdivided into groups by cutting the dendrogram at a desired similarity level.

### Principal Component Analysis (PCA)

**Purpose and Focus**:

* PCA is a dimensionality reduction technique. Its primary goal is to reduce the complexity of the data while retaining as much of the variation present in the original dataset as possible.
* It focuses on identifying the directions (principal components) in which the data varies most, essentially reorienting the data into a new set of coordinates to simplify and compress the dataset without significant loss of information.

**Methodology**:

* PCA works by calculating the eigenvectors and eigenvalues of the data’s covariance matrix. These eigenvectors represent the directions of maximum variance, and eigenvalues denote the magnitude of these directions.
* The resulting principal components are orthogonal to each other, ensuring that they capture distinct aspects of the data’s variability.

**Output**:

* The output of PCA is a set of principal components (new feature space) that are linear combinations of the original variables. These components are ranked based on their eigenvalues, with the first few components usually capturing the majority of the variation in the data.

### Clustering

**Purpose and Focus**:

* Clustering is a method of unsupervised learning used to group a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups.
* Its focus is on discovering the inherent groupings in the data, such as grouping customers by purchasing behavior or segmenting text documents with similar topics.

**Methodology**:

* Clustering algorithms classify objects into predefined groups based on similarity measures like Euclidean distance, Manhattan distance, or others. Popular methods include K-means clustering, hierarchical clustering, and DBSCAN.
* Unlike PCA, clustering does not involve transformation of the feature space. Instead, it seeks to identify partitions in the original data space, with each partition representing a cluster.

**Output**:

* The primary output of clustering is the cluster labels for each data point. These labels indicate the cluster membership of each data point, categorizing the dataset into distinct groups based on the similarity criteria defined.

### Key Differences

* **Transformation vs. Partitioning**: PCA transforms the feature space to reduce dimensionality, while clustering partitions the data into subsets based on similarity.
* **Output Interpretation**: PCA provides a transformed coordinate system where the most significant patterns in the data become more apparent. In contrast, clustering classifies data into different groups, making it useful for tasks like customer segmentation or identifying categories within data.
* **Use Case**: PCA is often used as a preprocessing step for other machine learning algorithms to improve performance by reducing overfitting and computational costs. Clustering is typically an end in itself, aiming to understand the structure or to extract insights from the data.

# Hands-on workshop: Principal Component Analysis and Clustering methods

**1. Principal Component Analysis (PCA)**

## Gentle Machine Learning  
## Principal Component Analysis  
  
  
# Dataset: USArrests is the sample dataset used in   
# McNeil, D. R. (1977) Interactive Data Analysis. New York: Wiley.  
# Murder numeric Murder arrests (per 100,000)  
# Assault numeric Assault arrests (per 100,000)  
# UrbanPop numeric Percent urban population  
# Rape numeric Rape arrests (per 100,000)  
# For each of the fifty states in the United States, the dataset contains the number   
# of arrests per 100,000 residents for each of three crimes: Assault, Murder, and Rape.   
# UrbanPop is the percent of the population in each state living in urban areas.  
library(datasets)  
library(ISLR)  
arrest = USArrests  
states=row.names(USArrests)  
names(USArrests)

## [1] "Murder" "Assault" "UrbanPop" "Rape"

# Get means and variances of variables  
apply(USArrests, 2, mean)

## Murder Assault UrbanPop Rape   
## 7.788 170.760 65.540 21.232

apply(USArrests, 2, var)

## Murder Assault UrbanPop Rape   
## 18.97047 6945.16571 209.51878 87.72916

# PCA with scaling  
pr.out=prcomp(USArrests, scale=TRUE)  
names(pr.out) # Five

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

pr.out$center # the centering and scaling used (means)

## Murder Assault UrbanPop Rape   
## 7.788 170.760 65.540 21.232

pr.out$scale # the matrix of variable loadings (eigenvectors)

## Murder Assault UrbanPop Rape   
## 4.355510 83.337661 14.474763 9.366385

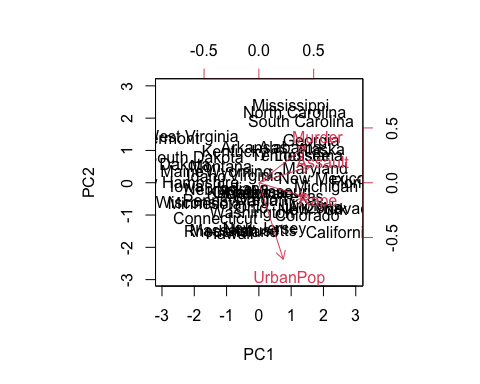
pr.out$rotation

## PC1 PC2 PC3 PC4  
## Murder -0.5358995 -0.4181809 0.3412327 0.64922780  
## Assault -0.5831836 -0.1879856 0.2681484 -0.74340748  
## UrbanPop -0.2781909 0.8728062 0.3780158 0.13387773  
## Rape -0.5434321 0.1673186 -0.8177779 0.08902432

dim(pr.out$x)

## [1] 50 4

pr.out$rotation=-pr.out$rotation  
pr.out$x=-pr.out$x  
biplot(pr.out, scale=0)



pr.out$sdev

## [1] 1.5748783 0.9948694 0.5971291 0.4164494

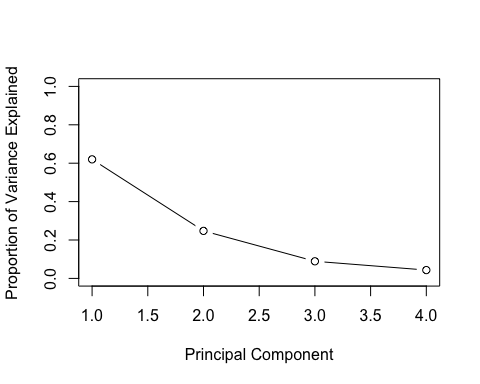
pr.var=pr.out$sdev^2  
pr.var

## [1] 2.4802416 0.9897652 0.3565632 0.1734301

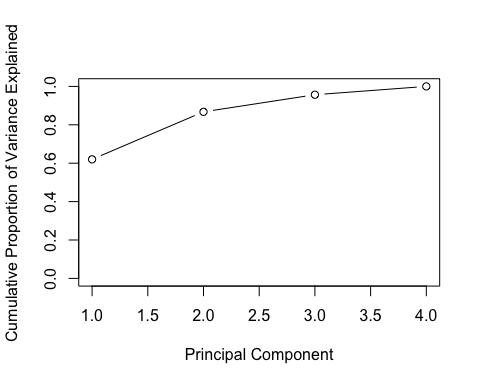
pve=pr.var/sum(pr.var)  
pve

## [1] 0.62006039 0.24744129 0.08914080 0.04335752

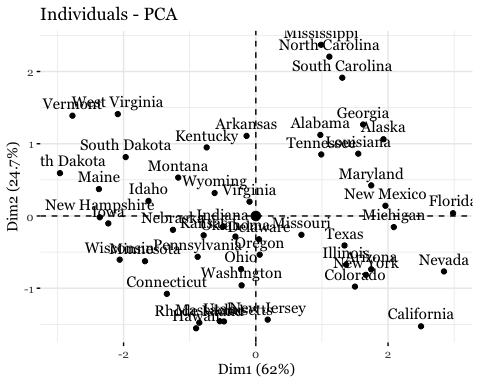
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1),type='b')



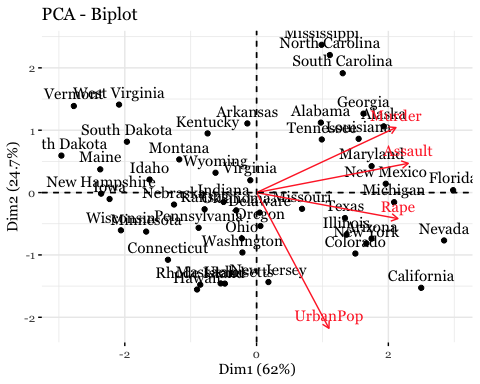
plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained", ylim=c(0,1),type='b')



## Use factoextra package  
library(factoextra)  
fviz(pr.out, "ind", geom = "auto", mean.point = TRUE, font.family = "Georgia")

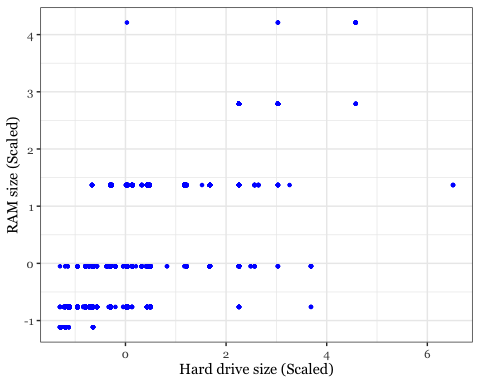


fviz\_pca\_biplot(pr.out, font.family = "Georgia", col.var="firebrick1")

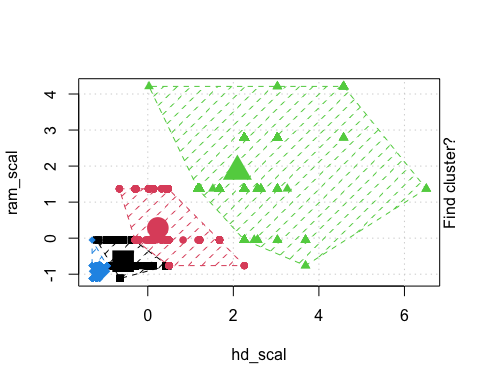
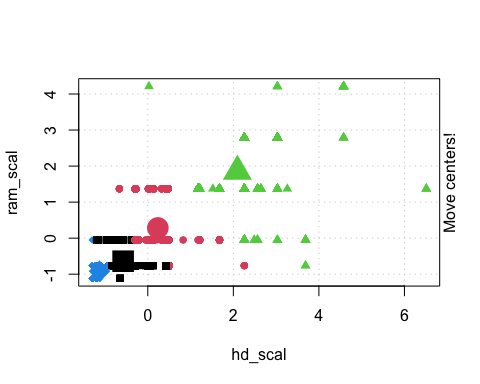
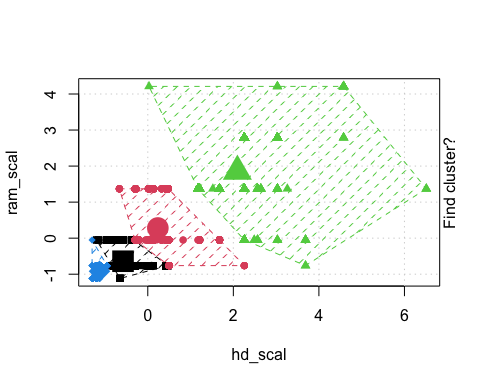
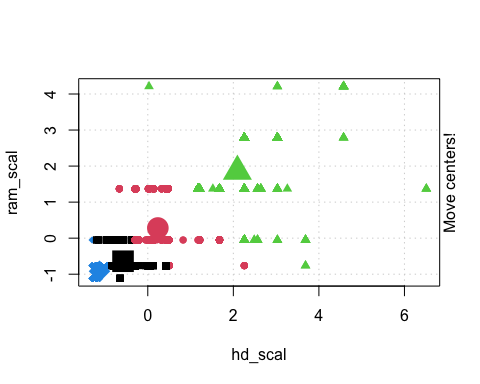
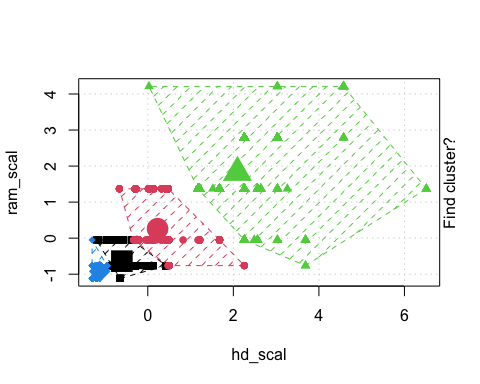
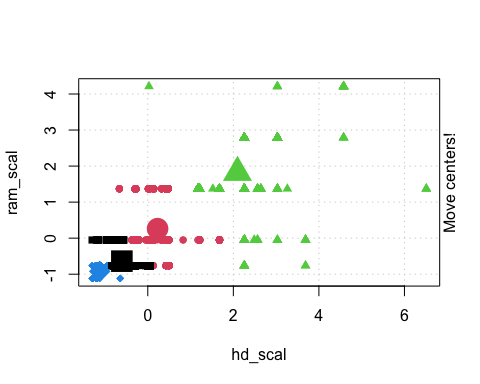
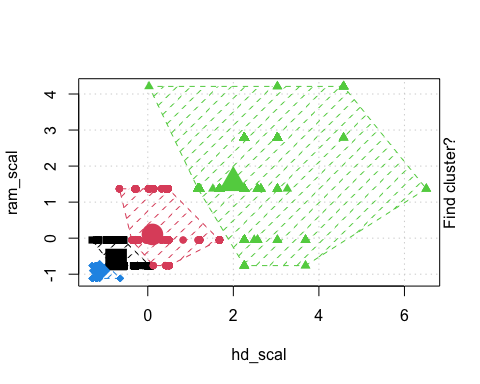
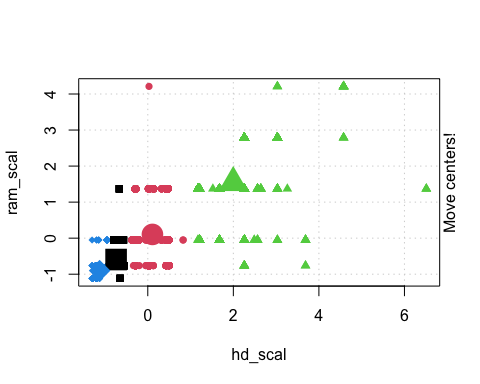
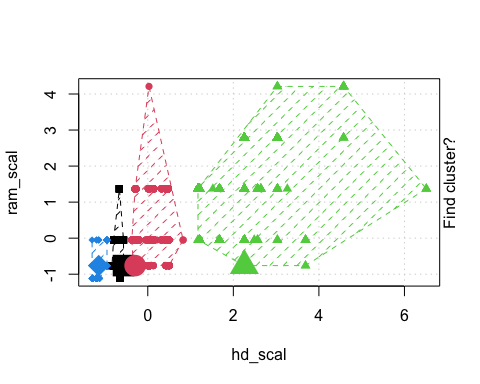
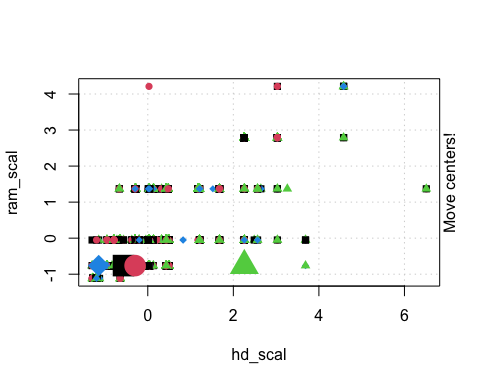


**2. K-Means Clustering**

## Computer purchase example: Animated illustration   
## Adapted from Guru99 tutorial (https://www.guru99.com/r-k-means-clustering.html)  
## Dataset: characteristics of computers purchased.  
## Variables used: RAM size, Harddrive size  
  
library(dplyr)  
library(ggplot2)  
library(RColorBrewer)  
  
computers = read.csv("https://raw.githubusercontent.com/guru99-edu/R-Programming/master/computers.csv")   
  
# Only retain two variables for illustration  
rescaled\_comp <- computers[4:5] %>%  
 mutate(hd\_scal = scale(hd),  
 ram\_scal = scale(ram)) %>%  
 select(c(hd\_scal, ram\_scal))  
   
ggplot(data = rescaled\_comp, aes(x = hd\_scal, y = ram\_scal)) +  
 geom\_point(pch=20, col = "blue") + theme\_bw() +  
 labs(x = "Hard drive size (Scaled)", y ="RAM size (Scaled)" ) +  
 theme(text = element\_text(family="Georgia"))

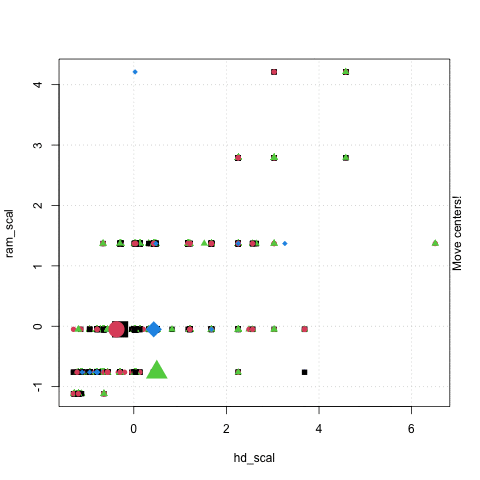


# install.packages("animation")  
library(animation)  
set.seed(2345)  
library(animation)  
  
# Animate the K-mean clustering process, cluster no. = 4  
kmeans.ani(rescaled\_comp[1:2], centers = 4, pch = 15:18, col = 1:4)



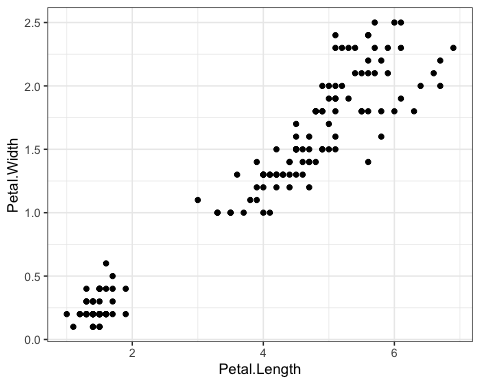
saveGIF(  
 kmeans.ani(rescaled\_comp[1:2], centers = 4, pch = 15:18, col = 1:4) ,  
 movie.name = "kmeans\_animated.gif",  
 img.name = "kmeans",  
 convert = "magick",  
 cmd.fun,  
 clean = TRUE,  
 extra.opts = ""  
)

## [1] TRUE

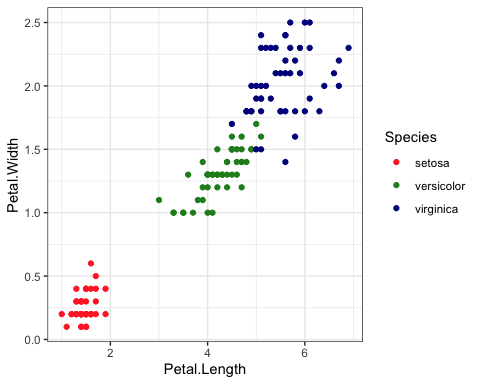


animated K-means output

## Iris example  
  
# Without grouping by species  
ggplot(iris, aes(Petal.Length, Petal.Width)) + geom\_point() +   
 theme\_bw() +  
 scale\_color\_manual(values=c("firebrick1","forestgreen","darkblue"))



# With grouping by species  
ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom\_point() +   
 theme\_bw() +  
 scale\_color\_manual(values=c("firebrick1","forestgreen","darkblue"))



# Check k-means clusters  
## Starting with three clusters and 20 initial configurations  
set.seed(20)  
irisCluster <- kmeans(iris[, 3:4], 3, nstart = 20)  
irisCluster

## K-means clustering with 3 clusters of sizes 50, 48, 52  
##   
## Cluster means:  
## Petal.Length Petal.Width  
## 1 1.462000 0.246000  
## 2 5.595833 2.037500  
## 3 4.269231 1.342308  
##   
## Clustering vector:  
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
## [75] 3 3 3 2 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 3 2 2 2 2  
## [112] 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2  
## [149] 2 2  
##   
## Within cluster sum of squares by cluster:  
## [1] 2.02200 16.29167 13.05769  
## (between\_SS / total\_SS = 94.3 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

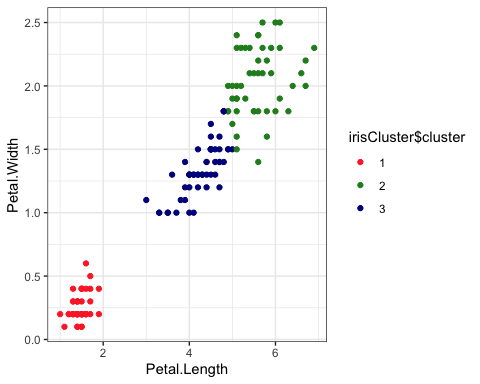
class(irisCluster$cluster)

## [1] "integer"

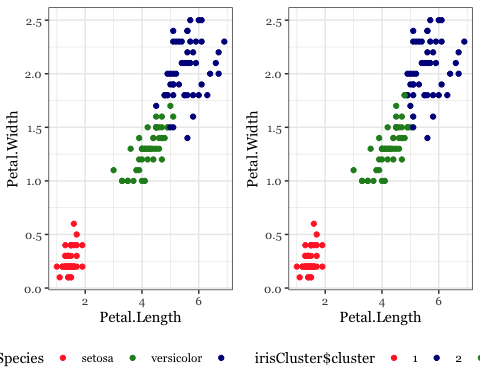
# Confusion matrix  
table(irisCluster$cluster, iris$Species)

##   
## setosa versicolor virginica  
## 1 50 0 0  
## 2 0 2 46  
## 3 0 48 4

irisCluster$cluster <- as.factor(irisCluster$cluster)  
ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) + geom\_point() +  
 scale\_color\_manual(values=c("firebrick1","forestgreen","darkblue")) +  
 theme\_bw()



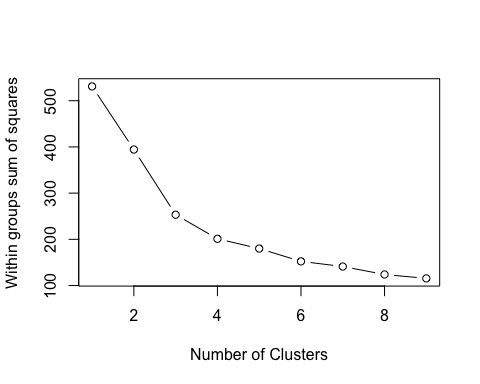
actual = ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom\_point() +   
 theme\_bw() +  
 scale\_color\_manual(values=c("firebrick1","forestgreen","darkblue")) +  
 theme(legend.position="bottom") +  
 theme(text = element\_text(family="Georgia"))   
kmc = ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) + geom\_point() +  
 theme\_bw() +  
 scale\_color\_manual(values=c("firebrick1", "darkblue", "forestgreen")) +  
 theme(legend.position="bottom") +  
 theme(text = element\_text(family="Georgia"))   
library(grid)  
library(gridExtra)  
grid.arrange(arrangeGrob(actual, kmc, ncol=2, widths=c(1,1)), nrow=1)



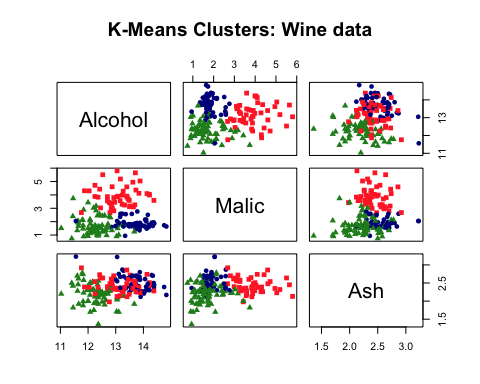
## Wine example  
  
# The wine dataset contains the results of a chemical analysis of wines   
# grown in a specific area of Italy. Three types of wine are represented in the   
# 178 samples, with the results of 13 chemical analyses recorded for each sample.   
# Variables used in this example:  
# Alcohol  
# Malic: Malic acid  
# Ash  
# Source: http://archive.ics.uci.edu/ml/datasets/Wine  
  
# Import wine dataset  
library(readr)  
wine <- read\_csv("https://raw.githubusercontent.com/datageneration/gentlemachinelearning/master/data/wine.csv")  
  
  
## Choose and scale variables  
wine\_subset <- scale(wine[ , c(2:4)])  
  
## Create cluster using k-means, k = 3, with 25 initial configurations  
wine\_cluster <- kmeans(wine\_subset, centers = 3,  
 iter.max = 10,  
 nstart = 25)  
wine\_cluster

## K-means clustering with 3 clusters of sizes 48, 60, 70  
##   
## Cluster means:  
## Alcohol Malic Ash  
## 1 0.1470536 1.3907328 0.2534220  
## 2 0.8914655 -0.4522073 0.5406223  
## 3 -0.8649501 -0.5660390 -0.6371656  
##   
## Clustering vector:  
## [1] 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2  
## [38] 2 3 1 2 1 2 1 3 1 1 2 2 2 3 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 2 3 3 2 2 2  
## [75] 3 3 3 3 3 1 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
## [112] 3 1 3 3 3 3 3 1 3 3 2 1 1 1 3 3 3 3 1 3 1 3 1 3 3 1 1 1 1 1 2 1 1 1 1 1 1  
## [149] 1 1 1 1 2 1 3 1 1 1 2 2 1 1 1 1 2 1 1 1 2 1 3 3 2 1 1 1 2 1  
##   
## Within cluster sum of squares by cluster:  
## [1] 73.71460 67.98619 111.63512  
## (between\_SS / total\_SS = 52.3 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

# Create a function to compute and plot total within-cluster sum of square (within-ness)  
wssplot <- function(data, nc=15, seed=1234){  
 wss <- (nrow(data)-1)\*sum(apply(data,2,var))  
 for (i in 2:nc){  
 set.seed(seed)  
 wss[i] <- sum(kmeans(data, centers=i)$withinss)}  
 plot(1:nc, wss, type="b", xlab="Number of Clusters",  
 ylab="Within groups sum of squares")  
}  
  
# plotting values for each cluster starting from 1 to 9  
wssplot(wine\_subset, nc = 9)



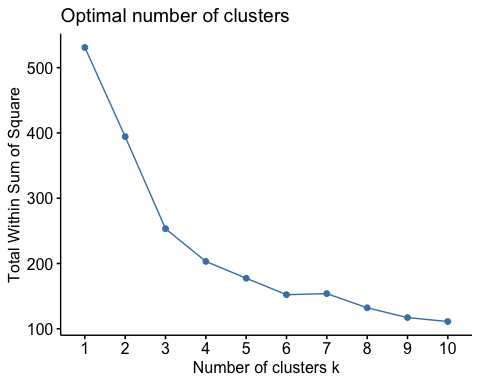
# Plot results by dimensions  
wine\_cluster$cluster = as.factor(wine\_cluster$cluster)  
pairs(wine[2:4],  
 col = c("firebrick1", "darkblue", "forestgreen")[wine\_cluster$cluster],  
 pch = c(15:17)[wine\_cluster$cluster],  
 main = "K-Means Clusters: Wine data")



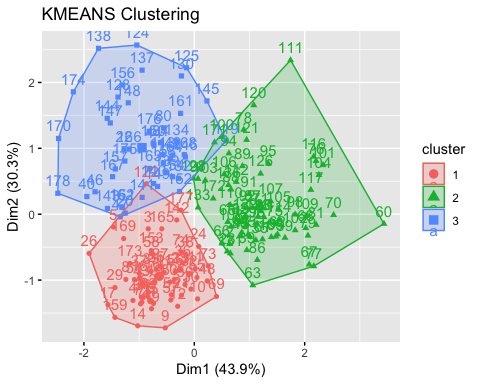
table(wine\_cluster$cluster)

##   
## 1 2 3   
## 48 60 70

## Use the factoextra package to do more  
# install.packages("factoextra")  
  
library(factoextra)  
fviz\_nbclust(wine\_subset, kmeans, method = "wss")



# Use eclust() procedure to do K-Means  
wine.km <- eclust(wine\_subset, "kmeans", nboot = 2)



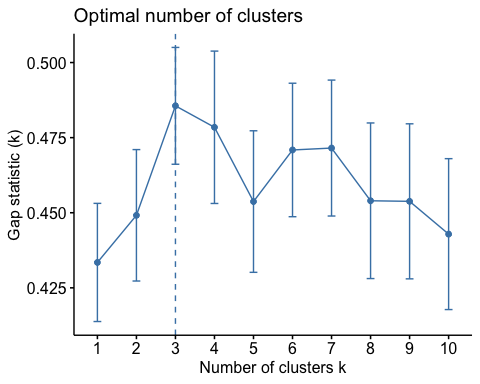
# Print result  
wine.km

## K-means clustering with 3 clusters of sizes 60, 70, 48  
##   
## Cluster means:  
## Alcohol Malic Ash  
## 1 0.8914655 -0.4522073 0.5406223  
## 2 -0.8649501 -0.5660390 -0.6371656  
## 3 0.1470536 1.3907328 0.2534220  
##   
## Clustering vector:  
## [1] 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 3 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1  
## [38] 1 2 3 1 3 1 3 2 3 3 1 1 1 2 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 2 2 1 1 1  
## [75] 2 2 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
## [112] 2 3 2 2 2 2 2 3 2 2 1 3 3 3 2 2 2 2 3 2 3 2 3 2 2 3 3 3 3 3 1 3 3 3 3 3 3  
## [149] 3 3 3 3 1 3 2 3 3 3 1 1 3 3 3 3 1 3 3 3 1 3 2 2 1 3 3 3 1 3  
##   
## Within cluster sum of squares by cluster:  
## [1] 67.98619 111.63512 73.71460  
## (between\_SS / total\_SS = 52.3 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault" "clust\_plot"   
## [11] "silinfo" "nbclust" "data" "gap\_stat"

# Optimal number of clusters using gap statistics  
wine.km$nbclust

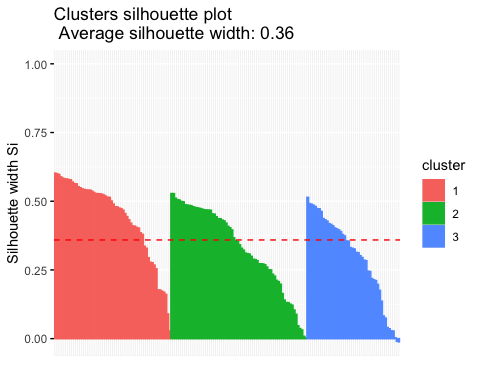
## [1] 3

fviz\_nbclust(wine\_subset, kmeans, method = "gap\_stat")

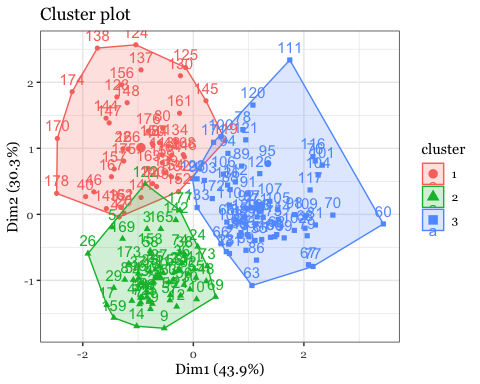


# Silhouette plot  
fviz\_silhouette(wine.km)

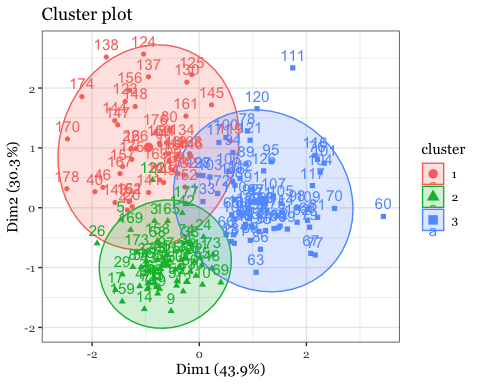
## cluster size ave.sil.width  
## 1 1 60 0.44  
## 2 2 70 0.33  
## 3 3 48 0.30



fviz\_cluster(wine\_cluster, data = wine\_subset) +   
 theme\_bw() +  
 theme(text = element\_text(family="Georgia"))

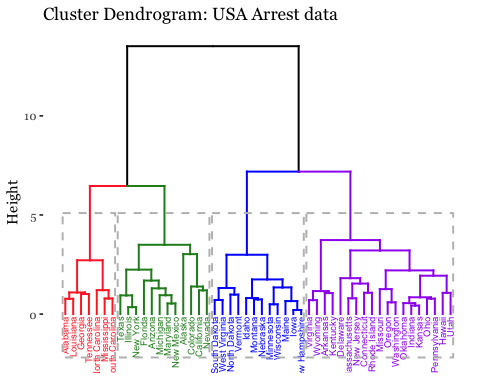


fviz\_cluster(wine\_cluster, data = wine\_subset, ellipse.type = "norm") +   
 theme\_bw() +  
 theme(text = element\_text(family="Georgia"))



**3. Hierarchical Clustering**

## Hierarchical Clustering  
## Dataset: USArrests  
# install.packages("cluster")  
arrest.hc <- USArrests %>%  
 scale() %>% # Scale all variables  
 dist(method = "euclidean") %>% # Euclidean distance for dissimilarity   
 hclust(method = "ward.D2") # Compute hierarchical clustering  
  
# Generate dendrogram using factoextra package  
fviz\_dend(arrest.hc, k = 4, # Four groups  
 cex = 0.5,   
 k\_colors = c("firebrick1","forestgreen","blue", "purple"),  
 color\_labels\_by\_k = TRUE, # color labels by groups  
 rect = TRUE, # Add rectangle (cluster) around groups,  
 main = "Cluster Dendrogram: USA Arrest data"  
) + theme(text = element\_text(family="Georgia"))



**References**

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013 *An introduction to statistical learning*. Vol. 112. New York: Springer.