UNIVERSITÉ DU LUXEMBOURG

LUXEMBOURG INSTITUTE OF HEALTH
RESEARCH DEDICATED TO LIFE

PhD Course

Advanced Biostatistics

Lecture 5

PCA, Clustering and Classification

dr. P. Nazarov

petr.nazarov@lih.lu

27-05-2016

Outline

- **Principal component analysis (L5.1)**
  - Iris dataset
  - General introduction to PCA
  - PCA for data exploratory analysis
- **Clustering (L5.2)**
  - k-means clustering
  - Hierarchical clustering
  - ALL dataset
- **Classification (L5.3)**
  - Overview
  - Feature characterization (ROC, AUC)
  - Support vector machines (SVM)

L5.1. Principal Component Analysis

Example: Iris Data (R.A.Fisher)

The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by Sir Ronald Aylmer Fisher (1936) as an example of discriminant analysis. It is sometimes called Anderson's Iris data set because Edgar Anderson collected the data to quantify the geographic variation of Iris flowers in the Gaspé Peninsula.
The dataset consists of 50 samples from each of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample, they are the length and the width of sepal and petal, in centimeters. Based on the combination of the four features, Fisher developed a linear discriminant model to distinguish the species from each other.

Iris setosa

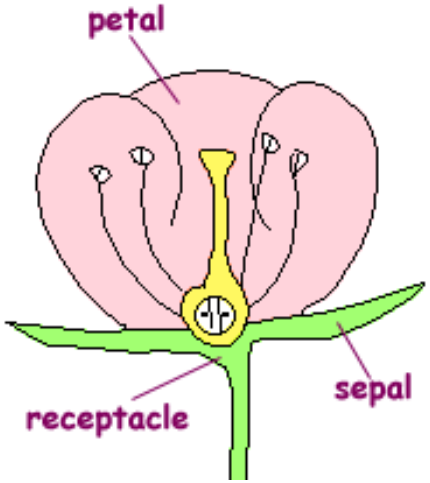
Iris versicolor

Iris virginica

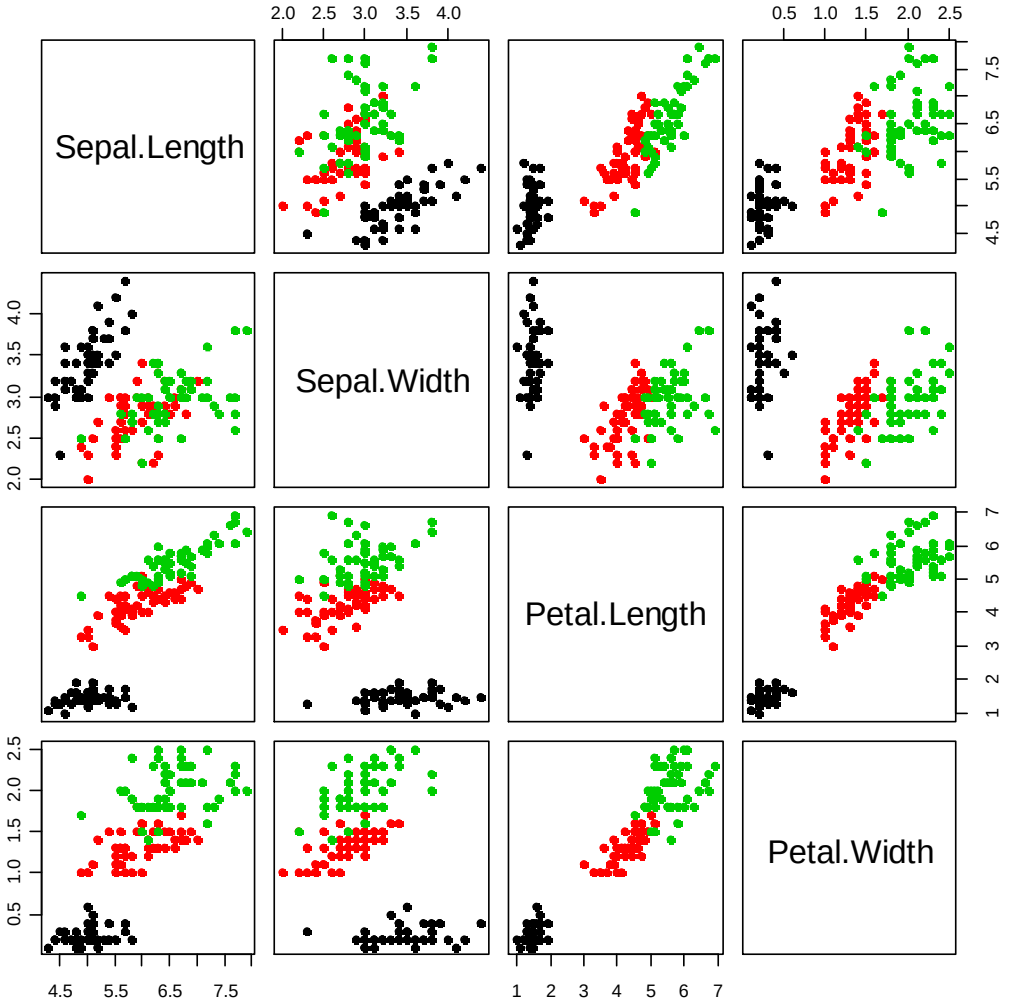
L5.1. Principal Component Analysis

Iris Dataset Presentation

```
iris
str(iris)
## plot iris data
x11()
plot(iris[,-5])
## more beautiful
plot(iris[,-5],
     col = iris[,5],pch=19)
```



http://urbanext.illinois.edu/gpe/case4/c4facts1a.html



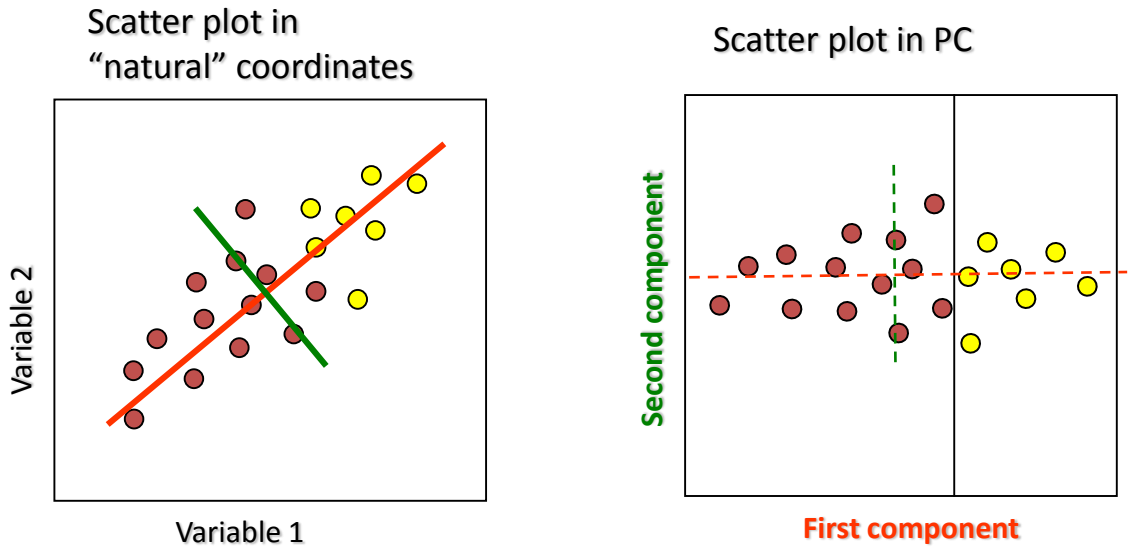
How could we possibly represent these data on a single plot?

Principal Component Analysis (PCA)

Principal component analysis (PCA)
is a vector space transform used to reduce multidimensional data sets to lower dimensions for analysis. It selects the **coordinates along which the variation of the data is bigger**.

20000 genes → 2 dimensions

For the simplicity let us consider 2 parametric situation both in terms of data and resulting PCA.



Instead of using 2 "natural" parameters for the classification, we can use the first component!

L5.1. Principal Component Analysis

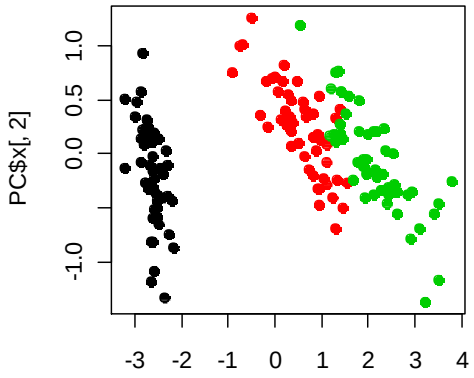
PCA for Iris Dataset

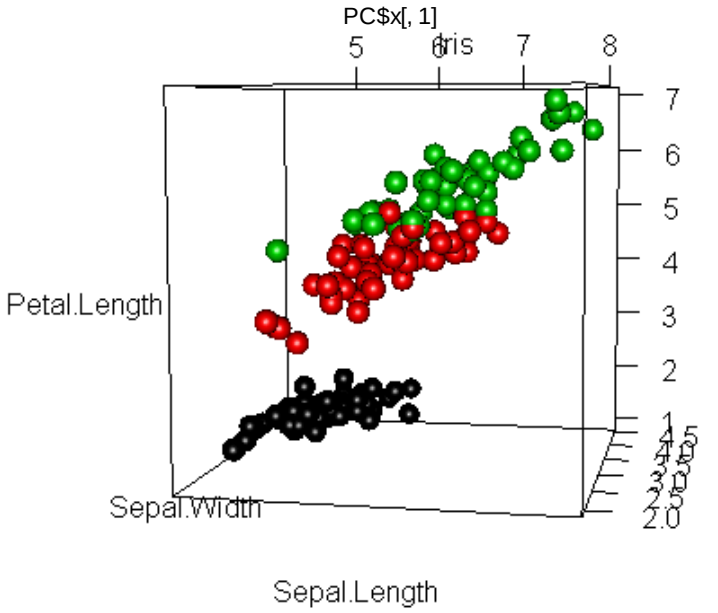
```
Data = as.matrix(iris[,-5])
row.names(Data) = as.character(iris[,5])
classes = as.integer(iris[,5])

## perform PCA
PC = prcomp(Data)
str(PC)
## plot PC1 and PC2 only
plot(PC$x[,1],PC$x[,2],
     col=classes, pch=19)
## plot 3D
library(rgl)
plot3d(PC$x[,1],PC$x[,2],PC$x[,3],
       size = 2,
       col  = classes,
       type = "s",
       xlab = "PC1",
       ylab = "PC2",
       zlab = "PC3")
```

Iris setosa (1)

Iris versicolor (2)

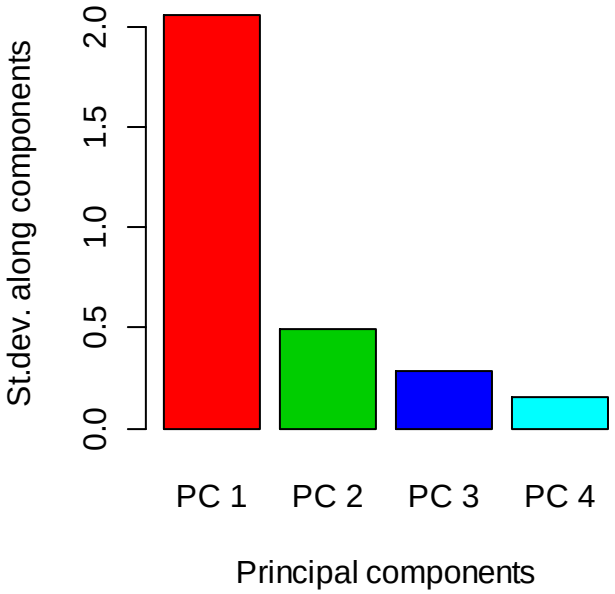
Iris virginica (3)

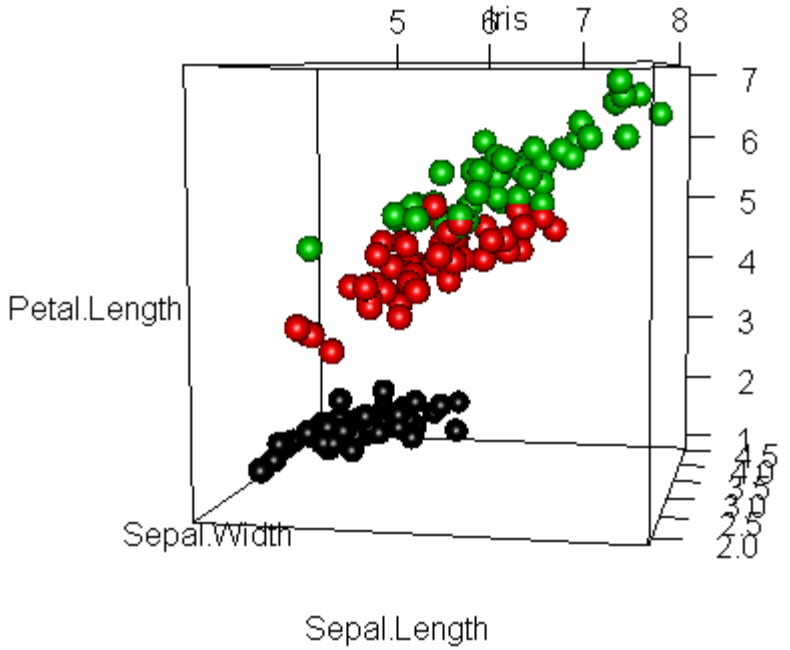




PCA for Iris Dataset

```
barplot(PC$sdev,
        names.arg = paste("PC",1:length(PC$sdev)),
        col   = (1:length(PC$sdev))+1,
        xlab = "Principal components",
        ylab = "St.dev. along components")
```

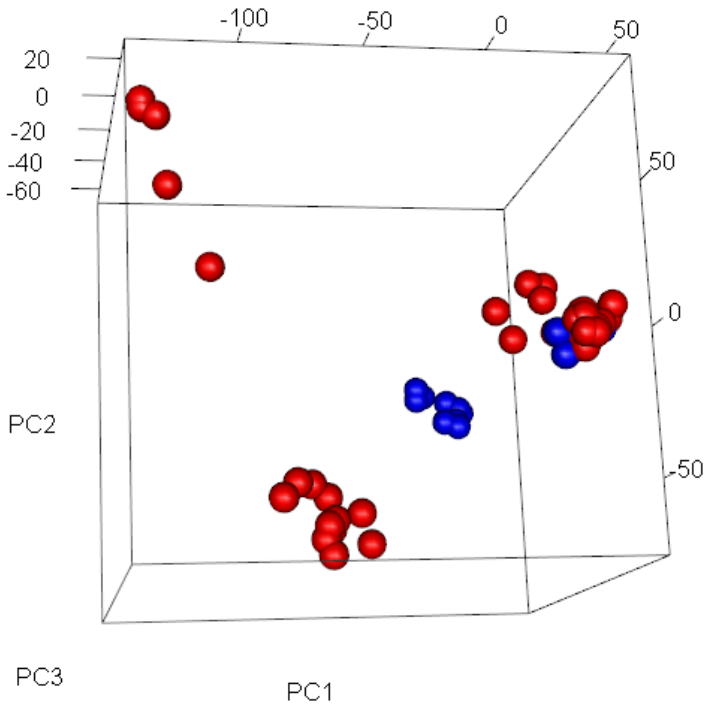




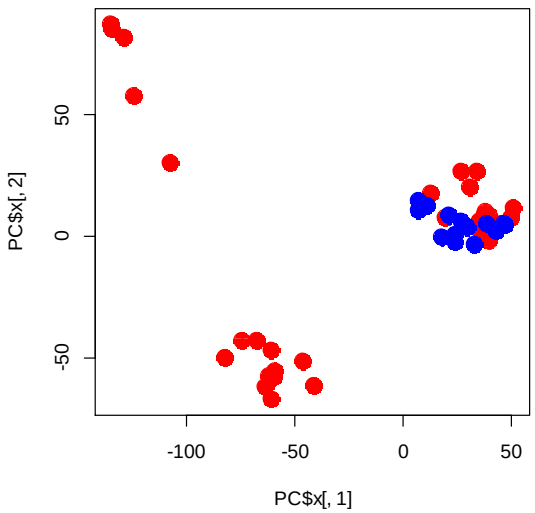
L5.1. Principal Component Analysis

Task: ALL data

Acute lymphoblastic leukemia (ALL), is a form of leukemia characterized by excess lymphoblasts. **all_data.txt** contains the results of full-trancript profiling for ALL patients and healthy donors using Affymetrix microarrays. The expression values in the table are in $\log_2$ scale. *Only 55 samples out of original 128 are kept (to have a 2-class problem)*



Perform a PCA analysis of the data



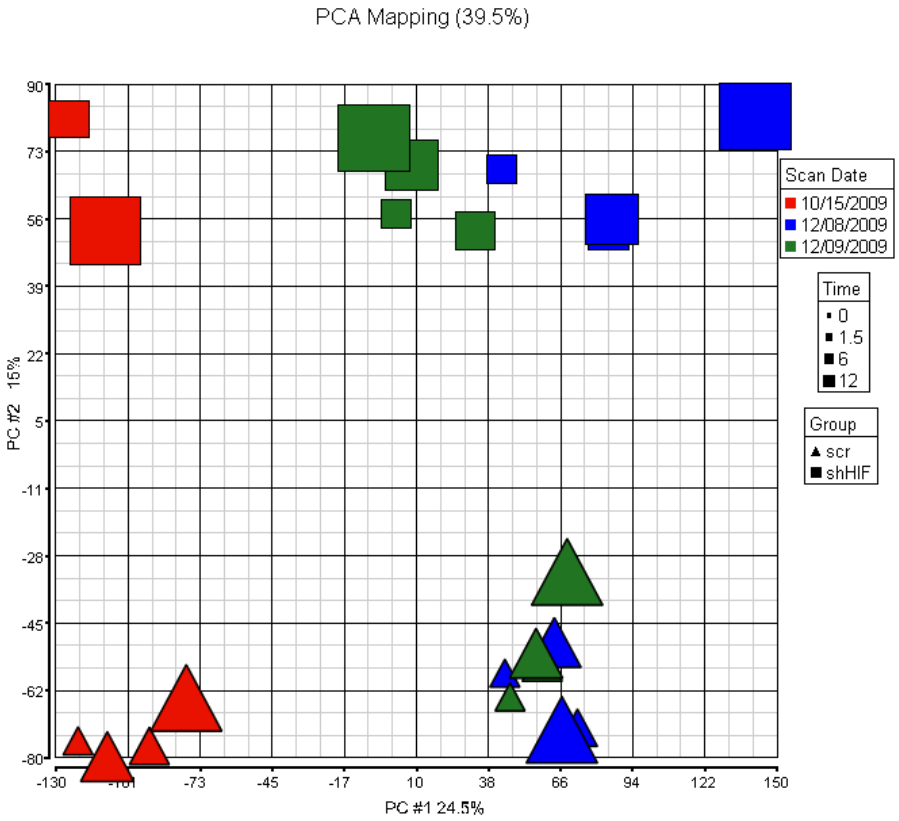
all_data.txt

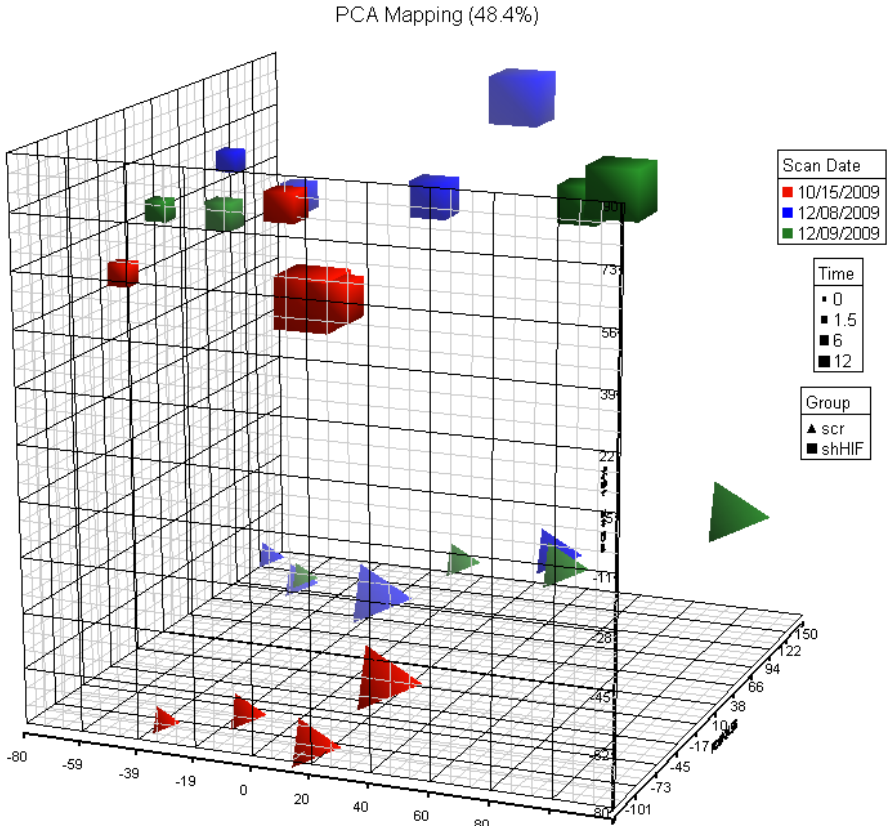
all_meta.txt

Chiaretti S., et al. Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. Blood, 2004, V.103

Exploratory Data Analysis

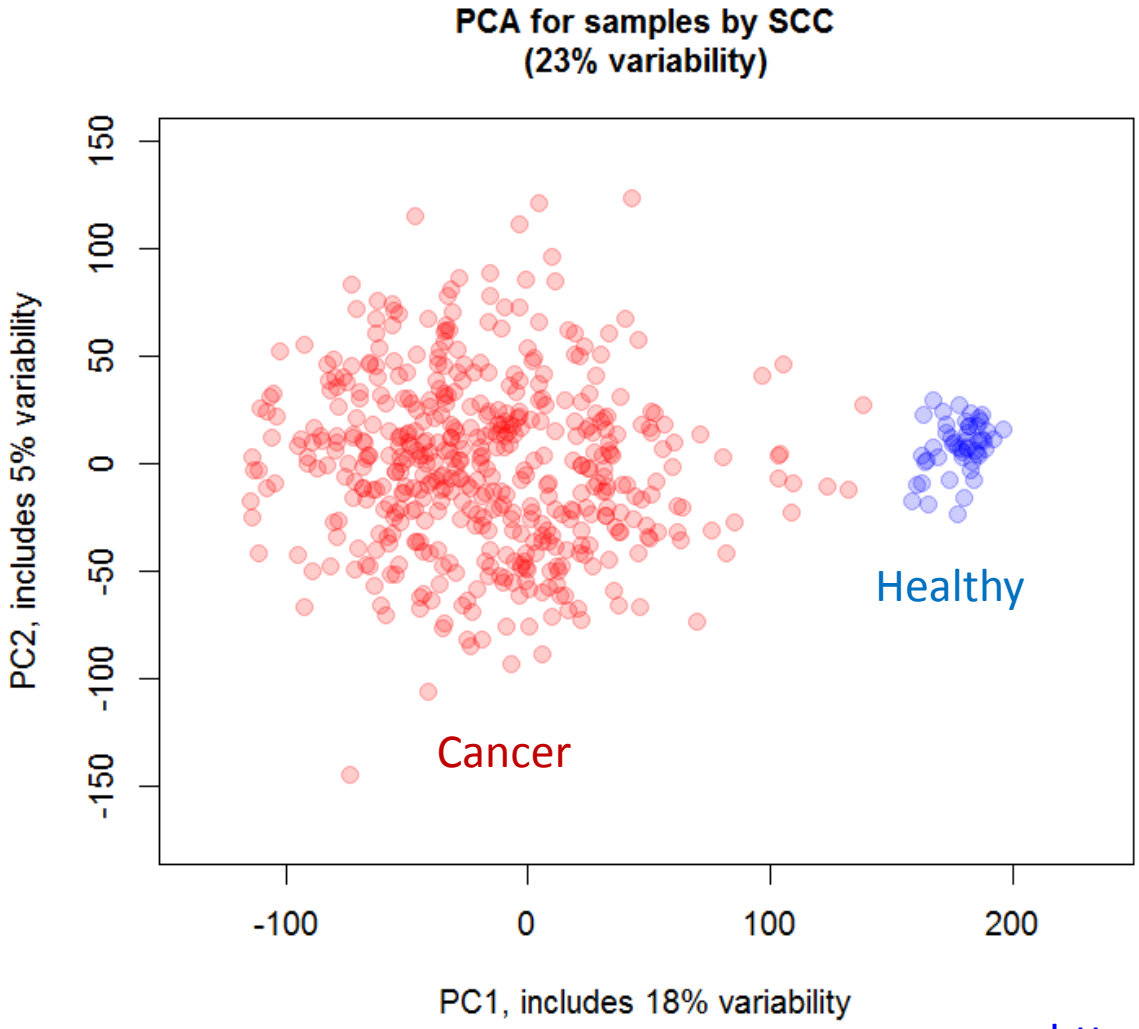
PCA





Exploratory Data Analysis

PCA in TCGA (LUSC data)



http://edu.sablab.net/data/txt/lusc.zip

L5.1. Principal Component Analysis

```
##############################################################################
# L5.1. PCA
##############################################################################
## clear memory
rm(list = ls())

##----------------------------------------------------------------------------
## L5.1.1. Iris data
##----------------------------------------------------------------------------

## show the data
iris
str(iris)

## plot iris data
x11()
plot(iris[,-5])
## more beautiful
plot(iris[,-5],col = iris[,5],pch=19)

##----------------------------------------------------------------------------
## L5.1.2 PCA
##----------------------------------------------------------------------------
## Let's transform data frame into numerical matrix
## data - numerical data
## classes - type of iris, 1=setosa, 2=versicolor, 3=virginica
Data = as.matrix(iris[,-5])
row.names(Data) = as.character(iris[,5])
classes = as.integer(iris[,5])

## perform PCA
PC = prcomp(Data)
str(PC)

## plot PC1 and PC2 only
plot(PC$x[,1],PC$x[,2],
     col=classes, pch=19)

## plot 3D
library(rgl)
plot3d(PC$x[,1],PC$x[,2],PC$x[,3],
       size = 2,
       col  = classes,
       type = "s",
       xlab = "PC1",
       ylab = "PC2",
       zlab = "PC3")
```

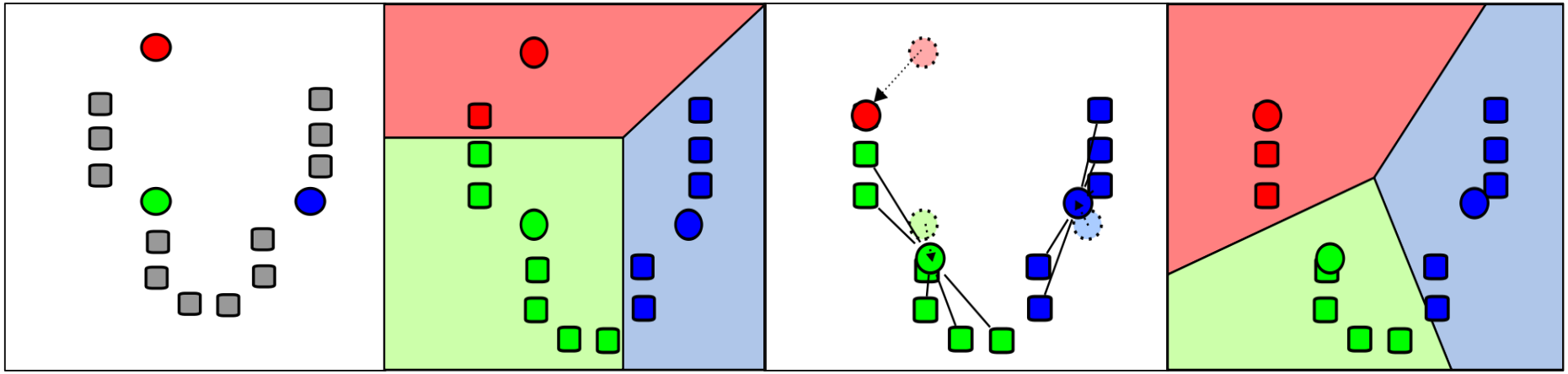
```
##----------------------------------------------------------------------------
## L5.1.2 PCA for ALL
##----------------------------------------------------------------------------

ALL = read.table("http://edu.sablab.net/data/txt/all_data.txt",
                 as.is=T,header=T,sep="\t")
## Transform to matrix
Data=as.matrix(ALL[,-(1:2)])
## assign colors based on column names
color = colnames(Data)
color[grep("ALL",color)]="red"
color[grep("normal",color)]="blue"
## perform PCA
PC = prcomp(t(Data))
## visualize in 3D
library(rgl)
plot3d(PC$x[,1],PC$x[,2],PC$x[,3],size = 2,col  = color,type = "s",
       xlab = "PC1",ylab = "PC2",zlab = "PC3")
## plot PC1 and PC2 only
plot(PC$x[,1],PC$x[,2],col=color,pch=19,cex=2)
text(PC$x[,1],PC$x[,2]+5,colnames(Data),cex=0.6)

## check the distribution of the data
source("http://sablab.net/scripts/plotDataPDF.r")
x11()
plotDataPDF(Data,col=color,add.legend=T)
x11()
boxplot(Data,col=color,outline=F,las=2)
```

Task L5.1

k-Means Clustering

k-Means Clustering

k-means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.



1) *k* initial "means" (in this case k=3) are randomly selected from the data set (shown in color).

2) *k* clusters are created by associating every observation with the nearest mean.

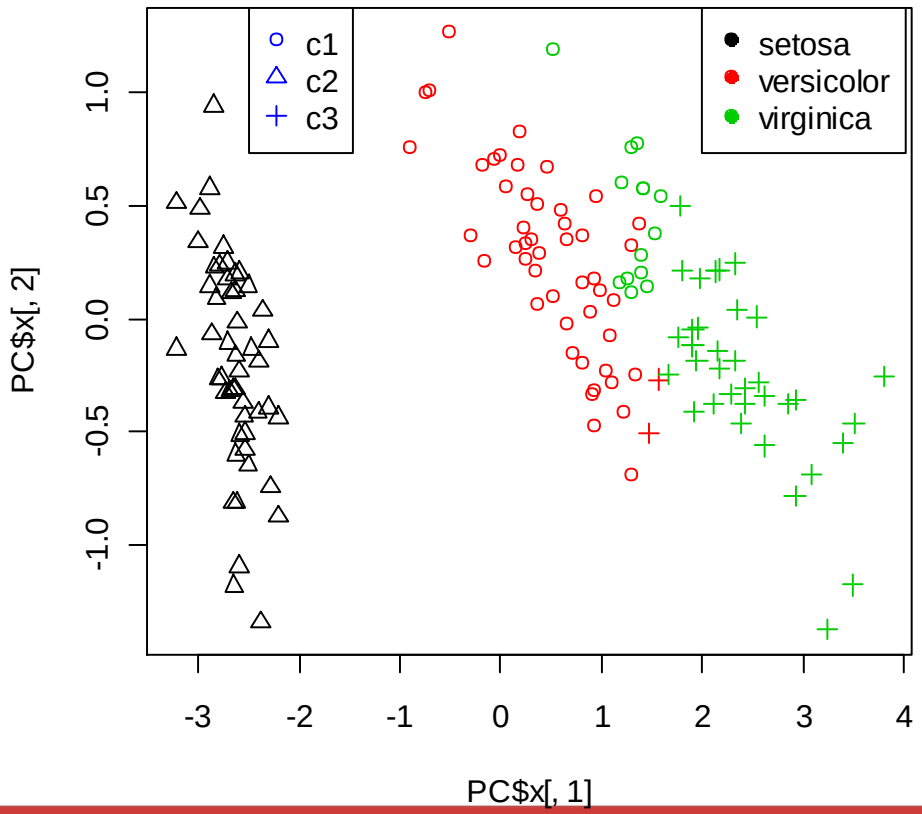
3) *The centroid of each of the k clusters becomes the new means.*

4) Steps 2 and 3 are repeated until convergence has been reached.

http://wikipedia.org

k-Means Clustering: Iris Dataset

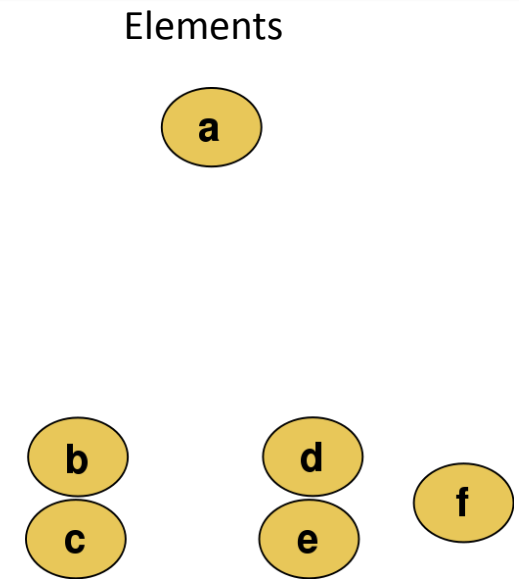
```
clusters = kmeans(x=Data,centers=3,nstart=10)$cluster
plot(PC$x[,1],PC$x[,2],col = classes,pch=clusters)
legend(2,1.4,levels(iris$Species),col=c(1,2,3),pch=19)
legend(-2.5,1.4,c("c1","c2","c3"),col=4,pch=c(1,2,3))
```

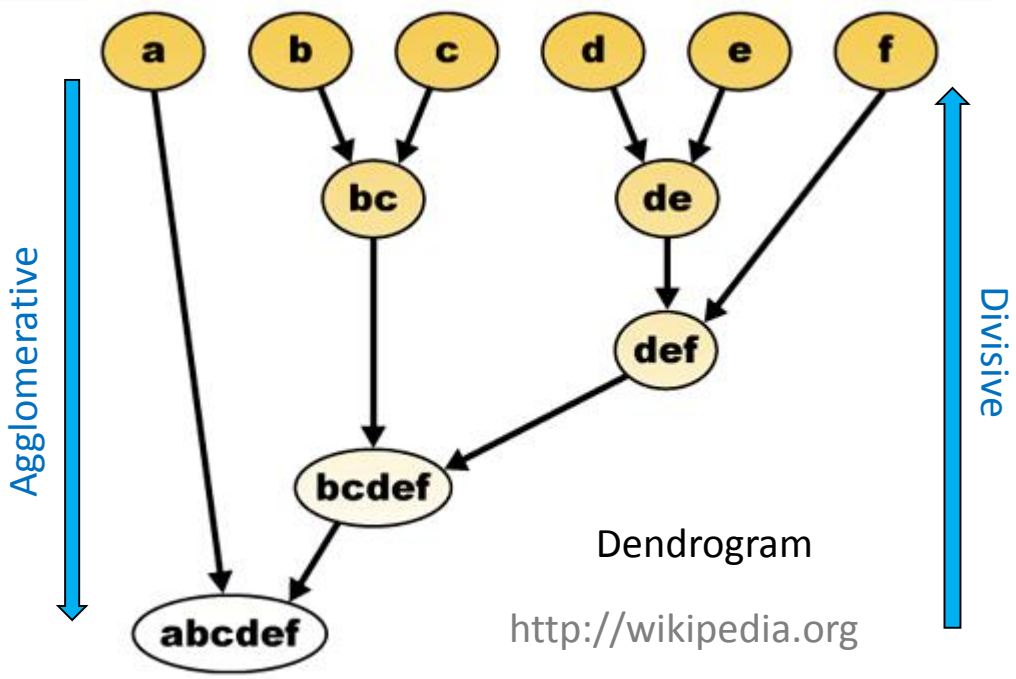


Hierarchical Clustering

Hierarchical Clustering

Hierarchical clustering creates a hierarchy of clusters which may be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all observations, and the leaves correspond to individual observations.
Algorithms for hierarchical clustering are generally either agglomerative, in which one starts at the leaves and successively merges clusters together; or divisive, in which one starts at the root and recursively splits the clusters.



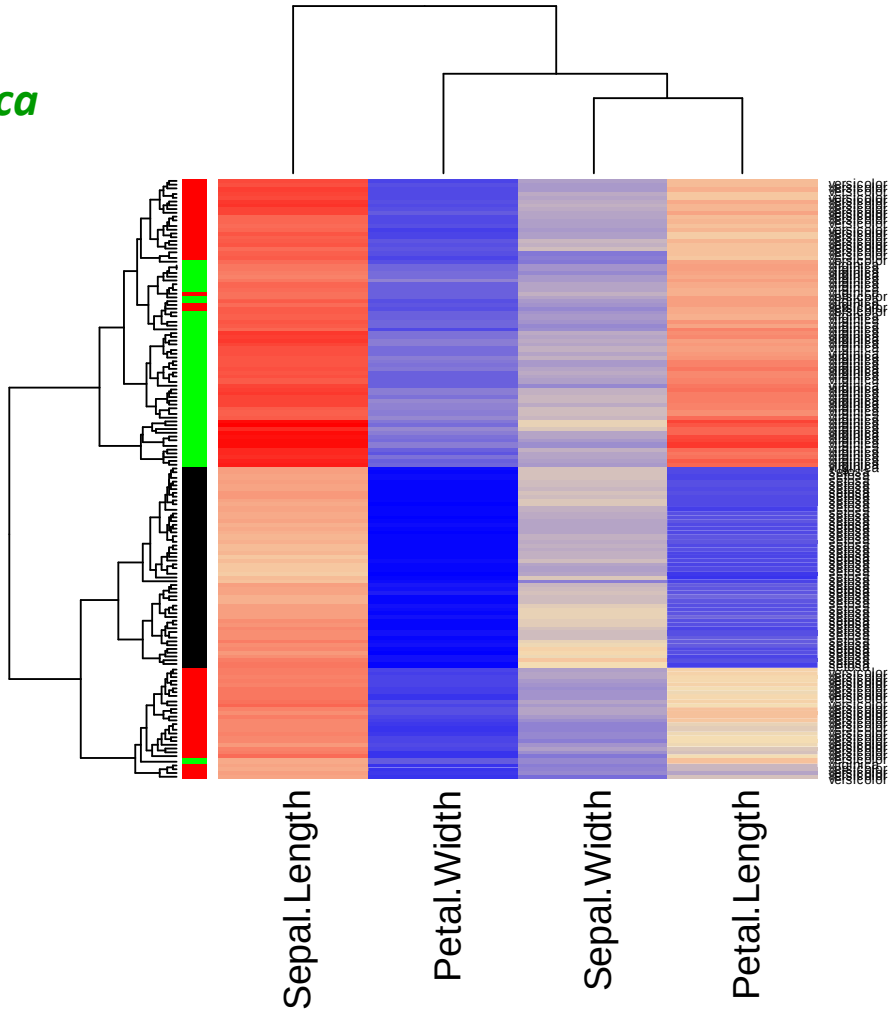


Hierarchical Clustering: Iris Dataset

Iris setosa

Iris versicolor

Iris virginica




```
## simple: use heatmap
heatmap(Data)

## use heatmap with colors
color = character(length(classes))
color[classes == 1] = "black"
color[classes == 2] = "red"
color[classes == 3] = "green"
heatmap(Data,RowSideColors=color,
        scale="none")

## modify the heatmap colors
heatmap(Data,RowSideColors=color,
    col = colorRampPalette (c("blue","wheat","red"))(1000))
```

Task: Clustering in ALL Dataset

1. Select the top 100 genes with the most significant variation of expression b/w normal and ALL samples. Use **`t.test(...)$p.val`** to address statistical significance.

2. Build a hierarchical clustering of the selected 100 genes and samples

```
ALL = read.table("http://edu.sablab.net/data/txt/all_data.txt",
                 as.is=T,header=T,sep="\t")
Data=as.matrix(ALL[,-(1:2)])
color = colnames(Data)
color[grep("ALL",color)]="red"
color[grep("normal",color)]="blue"
## annotate genes
rownames(Data) = ALL[,1]
## create indexes for normal and ALL columns
idx.norm = grep("normal",colnames(Data))
idx.all = grep("ALL",colnames(Data))
## perform a t-test
pv = double(nrow(Data))  ## p-value storage
for (i in 1:nrow(Data)){
  pv[i] = t.test(Data[i,idx.all],Data[i,idx.norm])$p.val
}
## select top genes
Top = Data[order(pv),][1:100,]
```

L5.2. Clustering


```
################################################################################
# L5.2. Clustering
################################################################################
## clear memory
rm(list = ls())

##-------------------------------------------------------------------------------
## L5.2.1 k-means Clustering
##-------------------------------------------------------------------------------
Data = as.matrix(iris[,-5])
row.names(Data) = as.character(iris[,5])
classes = as.integer(iris[,5])

## try k-means clustering
clusters = kmeans(x=Data,centers=3,nstart=10)$cluster

## show clusters on PCA
PC = prcomp(Data)
x11()
plot(PC$x[,1],PC$x[,2],col = classes,pch=clusters)
legend(2,1.4,levels(iris$Species),col=c(1,2,3),pch=19)
legend(-2.5,1.4,c("c1","c2","c3"),col=4,pch=c(1,2,3))

##-------------------------------------------------------------------------------
## L5.2.2 Hierarchical clustering
##-------------------------------------------------------------------------------

## use heatmap
heatmap(Data)

## use heatmap with colors
color = character(length(classes))
color[classes == 1] = "black"
color[classes == 2] = "red"
color[classes == 3] = "green"
heatmap(Data,RowSideColors=color,scale="none")

## modify the heatmap colors
heatmap(Data,RowSideColors=color,scale="none",
   col = colorRampPalette (c("blue","wheat","red"))(1000))

## For advanced heatmap use:
library(gplots)
heatmap.2(Data,RowSideColors=color,scale="none",trace="none",
          col = colorRampPalette (c("blue","wheat","red"))(1000))
```

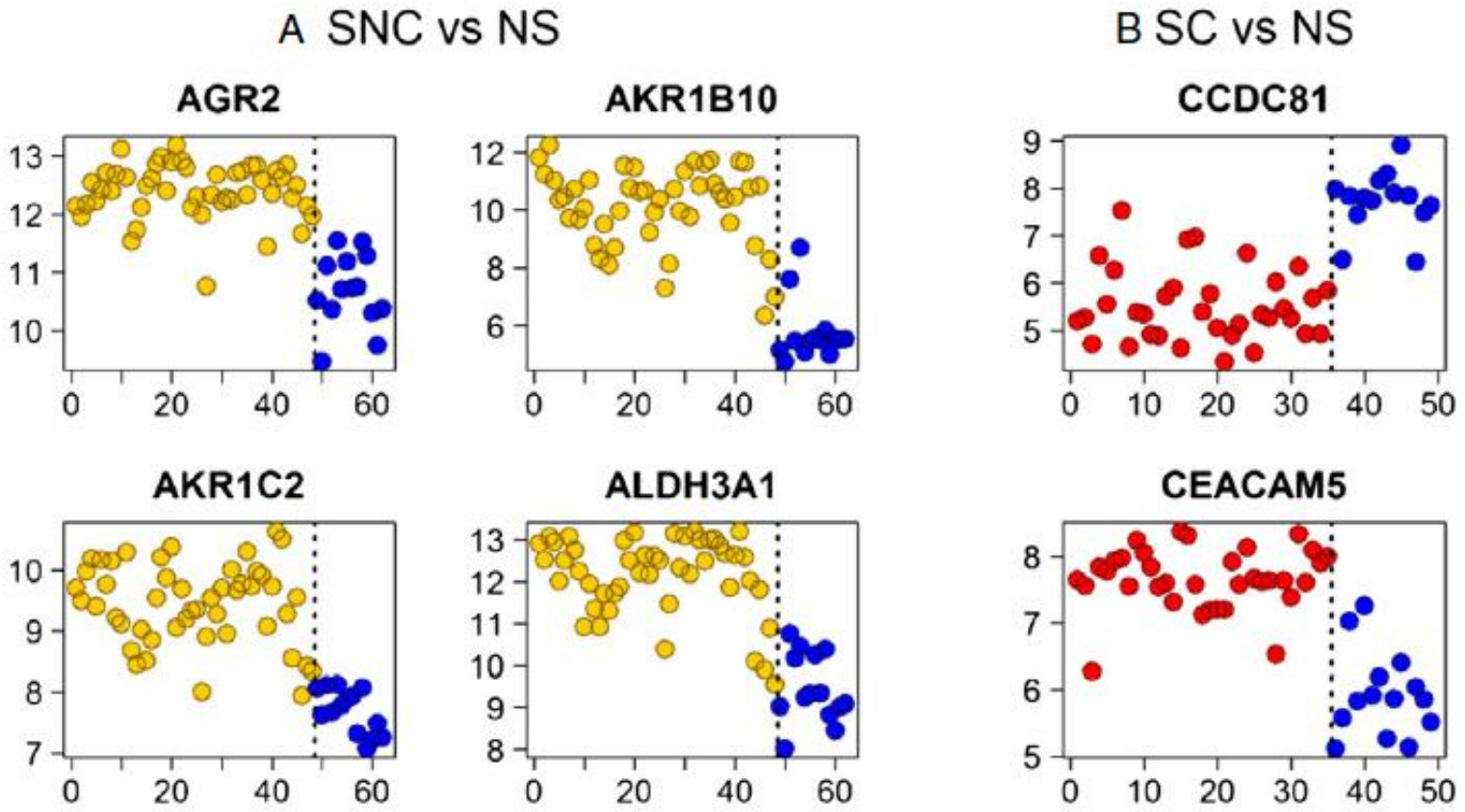
```
##-------------------------------------------------------------------------------
## L5.2.2 Hierarchical clustering: ALL (Task L5.2)
##-------------------------------------------------------------------------------
ALL = read.table("http://edu.sablab.net/data/txt/all_data.txt",
                 as.is=T,header=T,sep="\t")
Data=as.matrix(ALL[,-(1:2)])
color = colnames(Data)
color[grep("ALL",color)]="red"
color[grep("normal",color)]="blue"
##---------------------------------
## Task L5.2a. Select top 100 genes

## annotate genes
rownames(Data) = ALL[,1]
## create indexes for normal and ALL columns
idx.norm = grep("normal",colnames(Data))
idx.all = grep("ALL",colnames(Data))

## perform a t-test
pv = double(nrow(Data))
for (i in 1:nrow(Data)){
  pv[i] = t.test(Data[i,idx.all],Data[i,idx.norm])$p.val
}
## select top genes
Top = Data[order(pv),][1:100,]

## make a heatmap
heatmap(Top,ColSideColors=color,
          col = colorRampPalette (c("blue","wheat","red"))(1000))

## scale data first and repeat heatmap
TopSc = t(scale(t(Top)))
heatmap(TopSc,ColSideColors=color,
          col = colorRampPalette (c("blue","white","red"))(1000))


##---------------------------------
## In fact, robust set of signidicant genes, identified by limma
## package of R/Bioconductor is different:
x11()
source("http://sablab.net/scripts/limmaEBS2Class.r")
res = limmaEBS2Class(data = ALL[,-c(1:2)],
                     anno =  ALL[,c(1:2)],
                     classes = sub("_.+","",names(ALL[,-c(1:2)])),
                     plotTop=100,
                     col=c("red","blue"))
```

Task L5.2

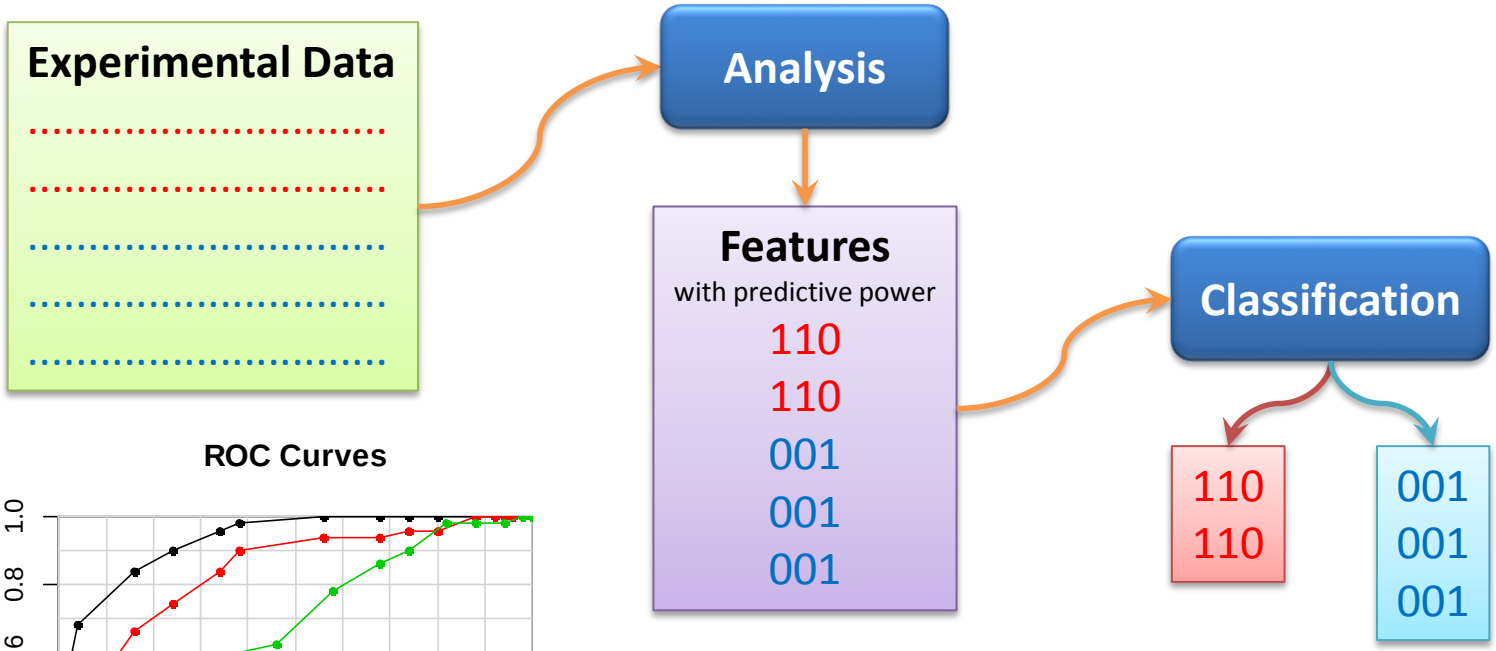
Gene Markers

Questions

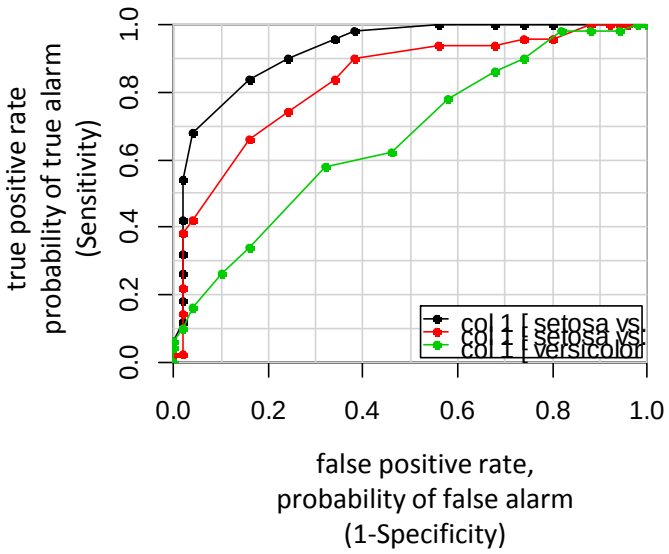
- Based on which genes or gene sets we can **predict** the group of the samples?
- How reliable is this prediction?



General Scheme



ROC Curves



Confusion Matrix

	A	B	C
pred.A	50	0	0
pred.B	0	48	2
pred.C	0	2	48

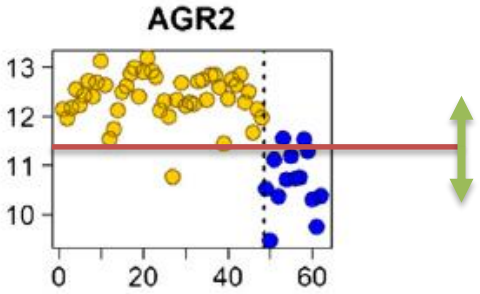
Selection of Features: ROC and AUC

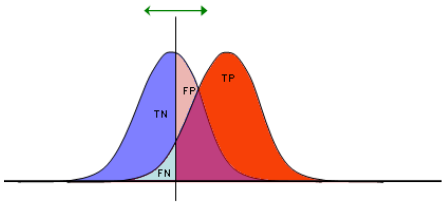
ROC curve
(receiver operating characteristic)
is a graphical plot of the sensitivity, or true positive rate, vs. false positive rate (1-specificity or false positive rate)

AUC
area under ROC curve: 1 – ideal separation, 0.5 – random separation.

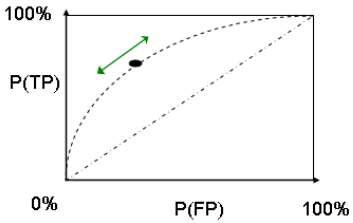
ROC is introduced for 2 classes.

If we have more then 2 classes – create several ROC curves (1 per class)

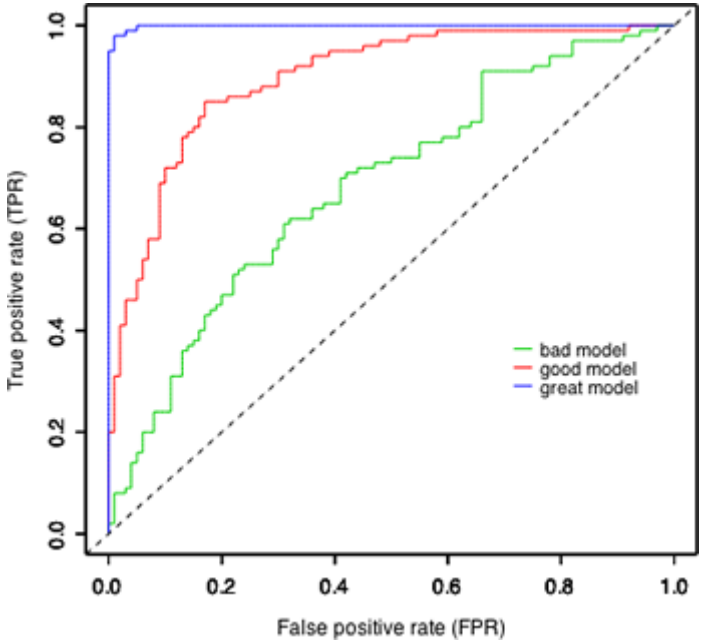




TP	FP
FN	TN
1	1



https://en.wikipedia.org/wiki/Receiver_operating_characteristic

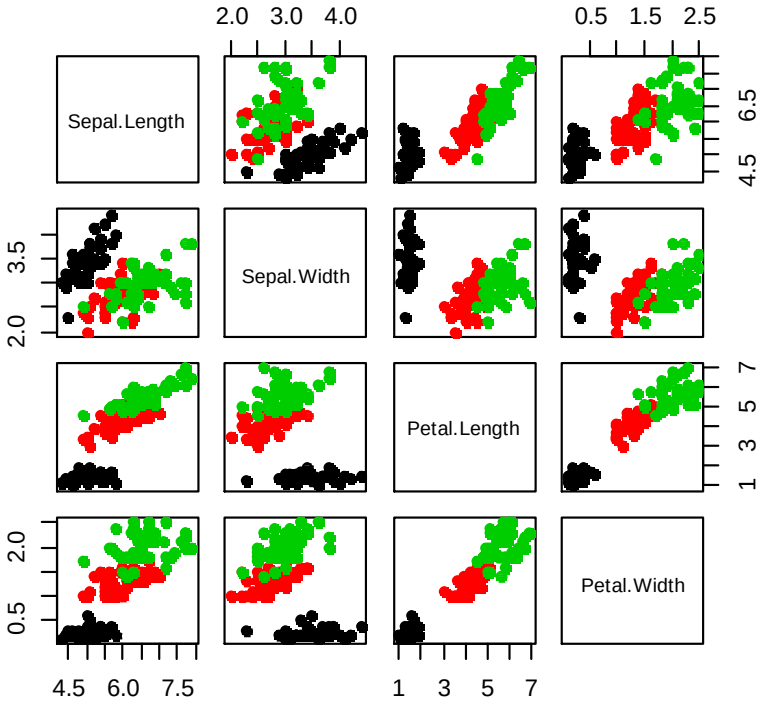


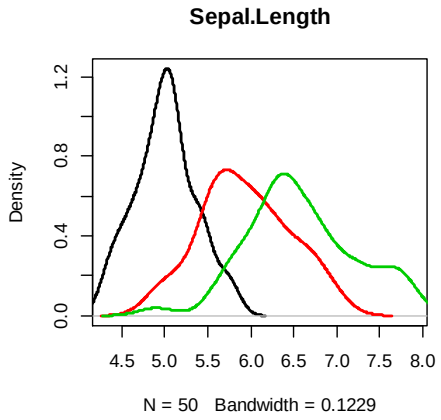
http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture22.htm

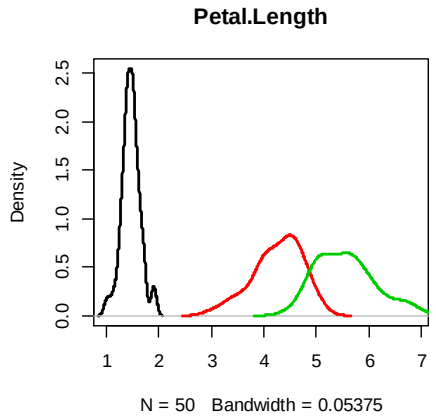
Selection of Features: Iris Dataset

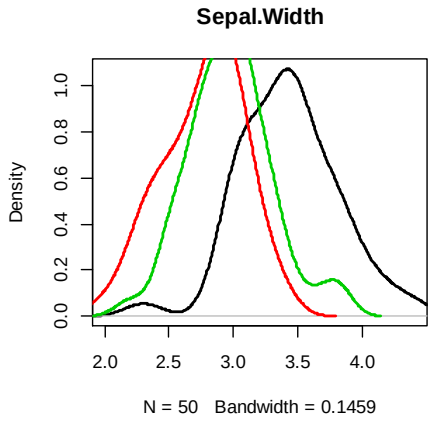
```
library(caTools)
plot(iris[,-5],col = iris[,5],pch=19)
## let's check which of the parameters is a better predictor
x11()
par(mfcol=c(2,2))
## plot probability densities
for (ipred in 1:4){
  plot(density(iris[as.integer(iris[,5])==1,ipred]),
       xlim=c(min(iris[,ipred]),max(iris[,ipred])),
       col=1,lwd=2,main=names(iris)[ipred])
  lines(density(iris[as.integer(iris[,5])==2,ipred]),col=2,lwd=2)
  lines(density(iris[as.integer(iris[,5])==3,ipred]),col=3,lwd=2)
}
x11()
par(mfcol=c(2,2))
## plot ROC curves and calculate AUC
for (ipred in 1:4){
  cat("\n\n",names(iris)[ipred],"\n")
  print(colAUC(iris[,ipred], iris[,5], plotROC=T))
}
```

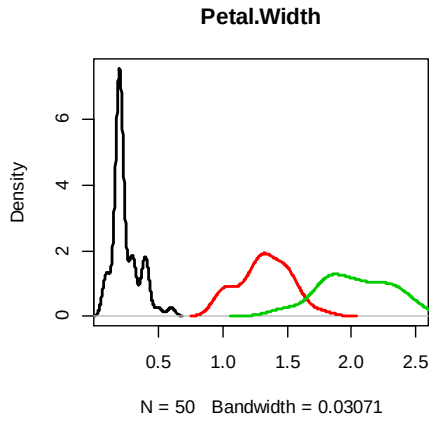
Selection of Features: Iris Dataset



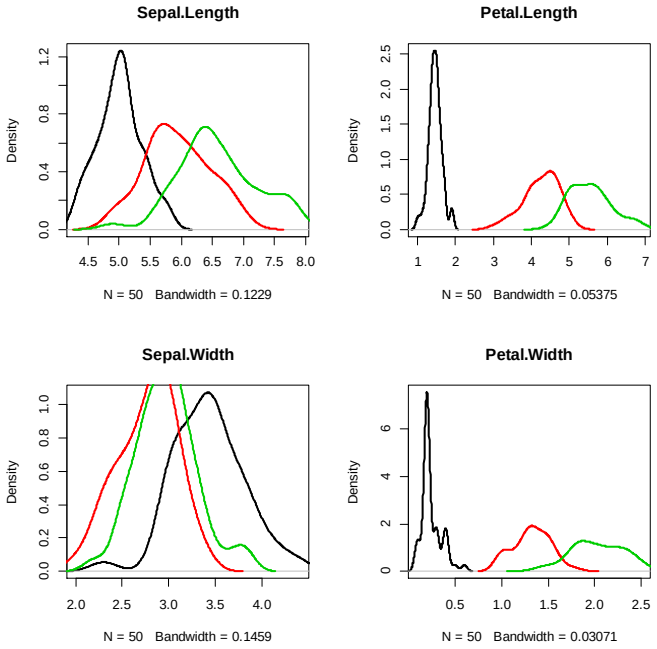


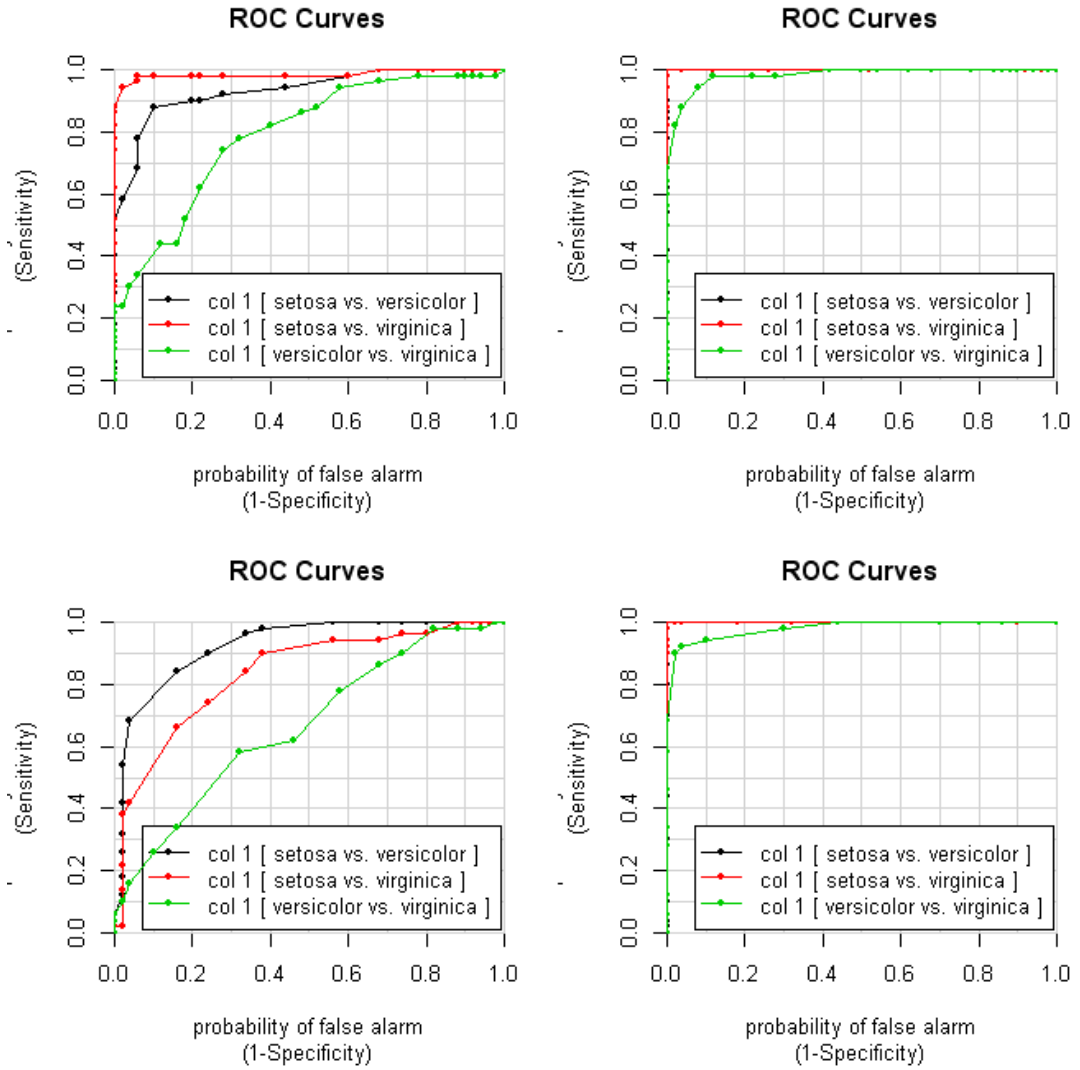






Selection of Features: Iris Dataset

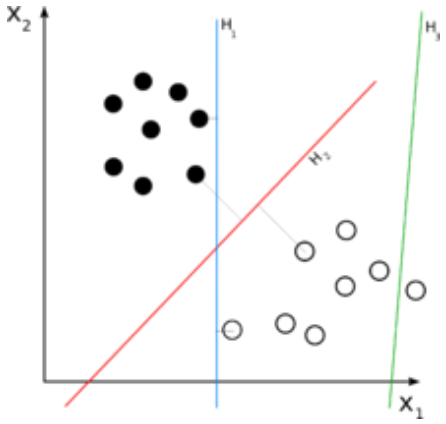




Example of Classification Methods

Support vector machine (SVM)

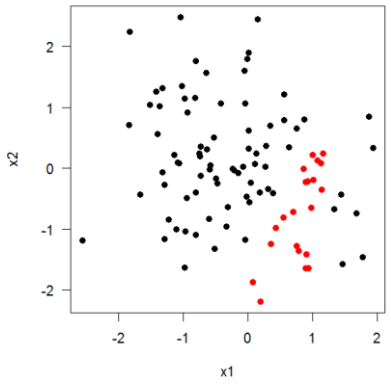
System tries to find a line (hyper plane) which
1) will divide you data to 2 groups, and
2) has the optimal distance from the closest elements of the groups



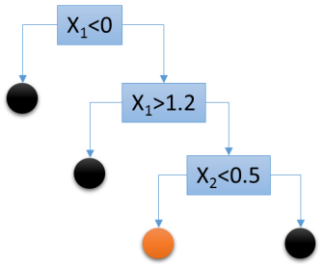
Random Forest (RF)

Makes a set of decision trees (if value x is less then x0 then we choose class A), which is called “forest”. Then vote among the trees.

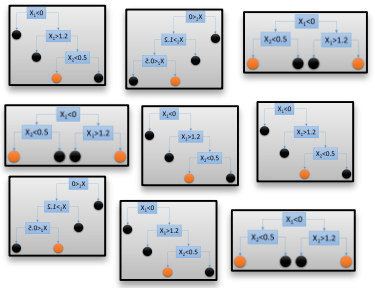
space of features



tree

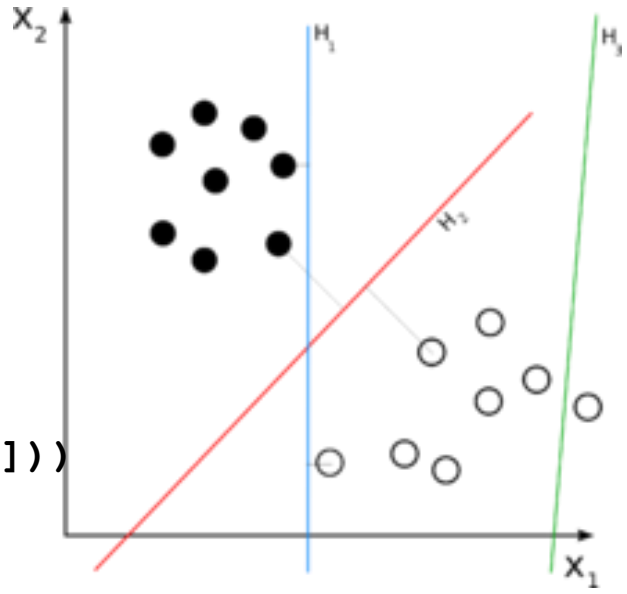


forest



SVM Classification

Support vector machine (SVM)
is a concept in statistics and computer science for a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis.




```
library(e1071)
model = svm(Species ~ ., data = iris)
svm.res = as.character(predict(model, iris[,-5]))

## creat a confusion matrix
ConTab = data.frame(matrix(nr=3,nc=3))
rownames(ConTab) = paste("pred.",levels(iris$Species),sep="")
names(ConTab) = levels(iris$Species)
for (ic in 1:3){
  for (ir in 1:3){
    ConTab[ir,ic] = sum(iris$Species == levels(iris$Species)[ic] &
            svm.res == levels(iris$Species)[ir])
  }
}
```

L5.3. Classification


```
################################################################################
# L5.3. Classification
################################################################################

##------------------------------------------------------------------------------
## L5.3.1 Iris data
##------------------------------------------------------------------------------

library(caTools)
plot(iris[,-5],col = iris[,5],pch=19)
## let's check which of the parameters is a better predictor
x11()
par(mfcol=c(2,2))
for (ipred in 1:4){
  plot(density(iris[as.integer(iris[,5])==1,ipred]),
       xlim=c(min(iris[,ipred]),max(iris[,ipred])),
       col=1,lwd=2,main=names(iris)[ipred])
  lines(density(iris[as.integer(iris[,5])==2,ipred]),col=2,lwd=2)
  lines(density(iris[as.integer(iris[,5])==3,ipred]),col=3,lwd=2)
}
x11()
par(mfcol=c(2,2))
for (ipred in 1:4){
  cat("\n\n",names(iris)[ipred],"\n")
  print(colAUC(iris[,ipred], iris[,5], plotROC=T))
}


library(e1071)
model = svm(Species ~ ., data = iris)
svm = as.character(predict(model, iris[,-5]))
res = cbind(as.character(iris[,5]), svm)

## creat a confusion matrix
ConTab = data.frame(matrix(nr=3,nc=3))
rownames(ConTab) = paste("pred.",levels(iris$Species),sep="")
names(ConTab) = levels(iris$Species)
for (ic in 1:3){
  for (ir in 1:3){
    ConTab[ir,ic] = sum(iris$Species == levels(iris$Species)[ic] & svm ==
levels(iris$Species)[ir])
  }
}
```

```
##------------------------------------------------------------------------------
## L5.3.1 Devaux et al. data
##------------------------------------------------------------------------------
## Devaux Y., et al. Use of circulating microRNAs to diagnose acute myocardial
## infarction. Clin Chem. 2012

MI = read.table("http://edu.sablab.net/data/txt/infarction.txt",
                header=T,sep="\t")

i.c  = MI$Type  == "ctrl"
i.ns = MI$Type  == "nstemi"
i.s  = MI$Type == "stemi"
color = character(nrow(MI))
color[i.c]="grey"
color[i.s]="green"
color[i.ns]="blue"

plot3d(MI[,2:4],type="s",size=1,col=color)
## to visualize overlaping samples we can but some noise
#plot3d(MI[,2:4]+rnorm(3*nrow(MI))*0.1,type="s",size=1,col=color)

model = svm(Type ~ ., data = MI)
svm= as.character(predict(model, MI[,-1]))
```

Task L5.3

Questions ?

Thank you for your attention

to be continued…