

Laboratorio 4 Bioinformática - Expresión génica

Manuel Villalobos Cid

15 de octubre de 2015

1. Introducción.

Incluso células que poseen el mismo ADN son diferentes entre sí debido a que presentan distinta expresión génica. Este proceso es conocido como el **Dogma Central de la Biología Molecular**: un gen expresa su información por transcripción de ADN en ARN mensajero y es traducido a una proteína.

El estudio de expresión génica y análisis de mutaciones permite a los científicos comprender los mecanismos fisiopatológicos de las enfermedades para así determinar sus tratamientos. Esto se logra estableciendo diferencias de expresión entre muestras normales y patológicas. También ha sido utilizado para estudiar características poblacionales o determinar el efecto de medicamentos en células blanco.

Los microarreglos son una de las técnicas mas utilizadas para establecer perfiles de transcripción. Existen múltiples desarrollos comerciales y académicos que presentan características diversas en cobertura, disponibilidad, especificidad y sensibilidad. Los arreglos fabricados por *Affymetrix* son los mayormente utilizados. Sin embargo, también se emplean los contruidos por *Agilent* o los de uso académico *CATMA*. Una de las principales fuentes de datos de microarreglos es [Gene Expression Omnibus de NCBI \(GEO\)](#).

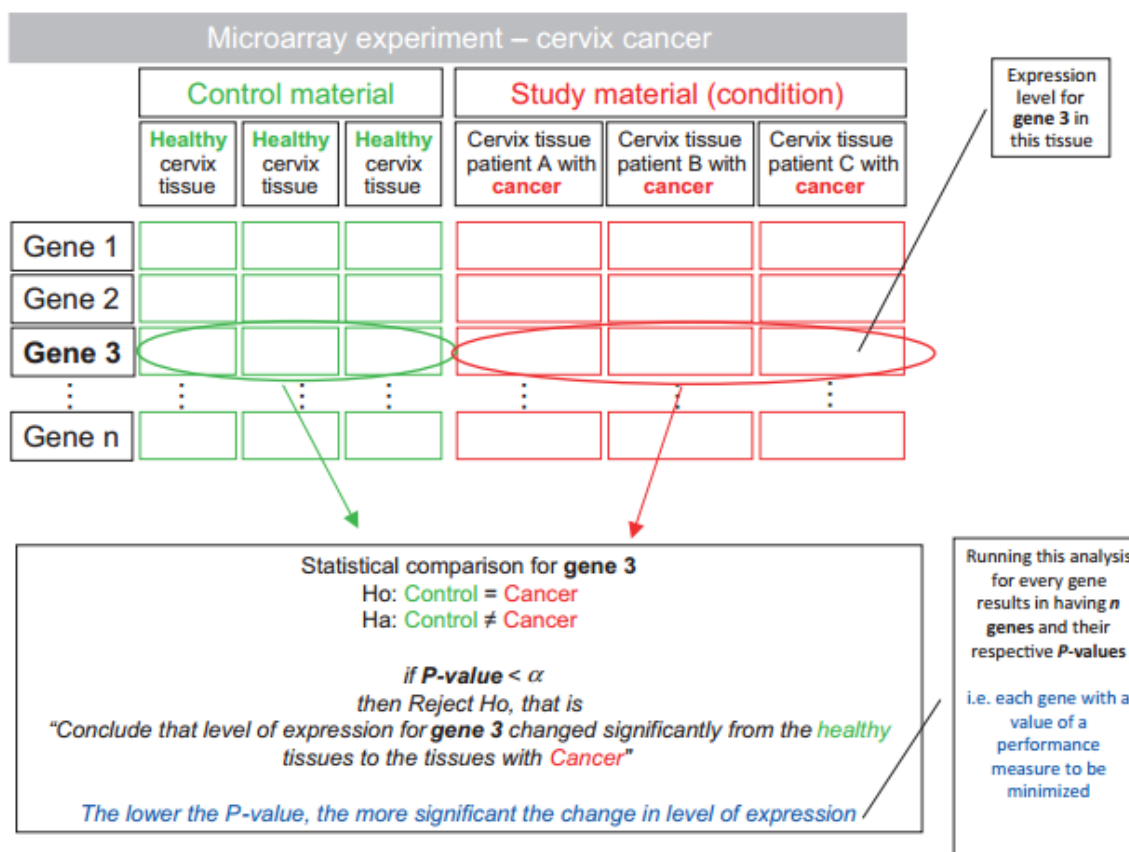


Figure 1.- Esquema de experimento para análisis de expresión génica con microarreglos.

Un juego de datos se caracteriza por tener una alta dimensionalidad: alrededor de 10.000 genes con más de 10 experimentos, además puede presentar ruido o deficiencias provenientes del proceso de hibridación. A raíz de este problema han surgido otras técnicas, como el Análisis en Serie de Expresión de Genes (Serial Analysis of Gene Expression, *SAGE*) o Firma Paralela de Secuenciación Masiva (Massively Parallel Signature Sequencing, *MPSS*). Ambas han demostrado ser mas robustas que el uso de microarreglos, ya que sus resultados no dependen de la selección de la sonda inicial. Recientemente se ha desarrollado una técnica denominada *RNA-seq*, cuyos resultados tampoco dependen de selección de sonda y no posee sesgo producido durante el proceso de hibridación. Sin embargo, aún no existe un conjunto de herramientas definitivas para procesar estos datos y el proceso de preparación de muestras continúa siendo bastante lento.

Para encontrar medidas de similaridad en expresión de genes se utiliza el concepto de distancia. Las más utilizadas corresponden a la Euclidiana y la correlación de *Pearson*. Sobre estas medidas se aplica minería de datos con métodos de clasificación y agrupamiento. Algoritmos basados en *UPGMA*, *NJ*, Mapas Auto-organizados (Self Organizing MAPS, *SOM*), lógica difusa, *SVM* y redes neuronales han sido ampliamente utilizados. Los algoritmos se pueden dividir en métodos planos, jerárquicos, basados en grafos, metaheurísticos y orientados a optimización. Éstos últimos pueden combinar varios de los grupos anteriores.

Esta guía de laboratorio tiene como objetivo que los alumnos se familiaricen con los conceptos relacionados con expresión génica, efectuando una pequeña experiencia en R, usando librerías disponibles por Bioconductor.

2. Actividades prácticas

2.1 Gene Expression Omnibus: búsqueda, descarga y análisis de conjunto de datos

- Acceda a la base de datos de [Gene Expression Omnibus de NCBI \(GEO\)](#).
- Puede buscar cualquier conjunto de datos utilizando el nombre de una determinada patología, publicación o identificador. Por ejemplo, buscar el conjunto de datos [GDS3709](#) y descargarlo haciendo click en **DataSet SOFT file**.
- Puede revisar los resultados ya obtenidos para los conjunto de datos, accediendo al **clustergrama** ubicado a la derecha de la pantalla. También puede utilizar las herramientas denominadas **Data Analysis Tools**.
- Investigue como funciona este juego de herramientas: **Data Analysis Tools**.

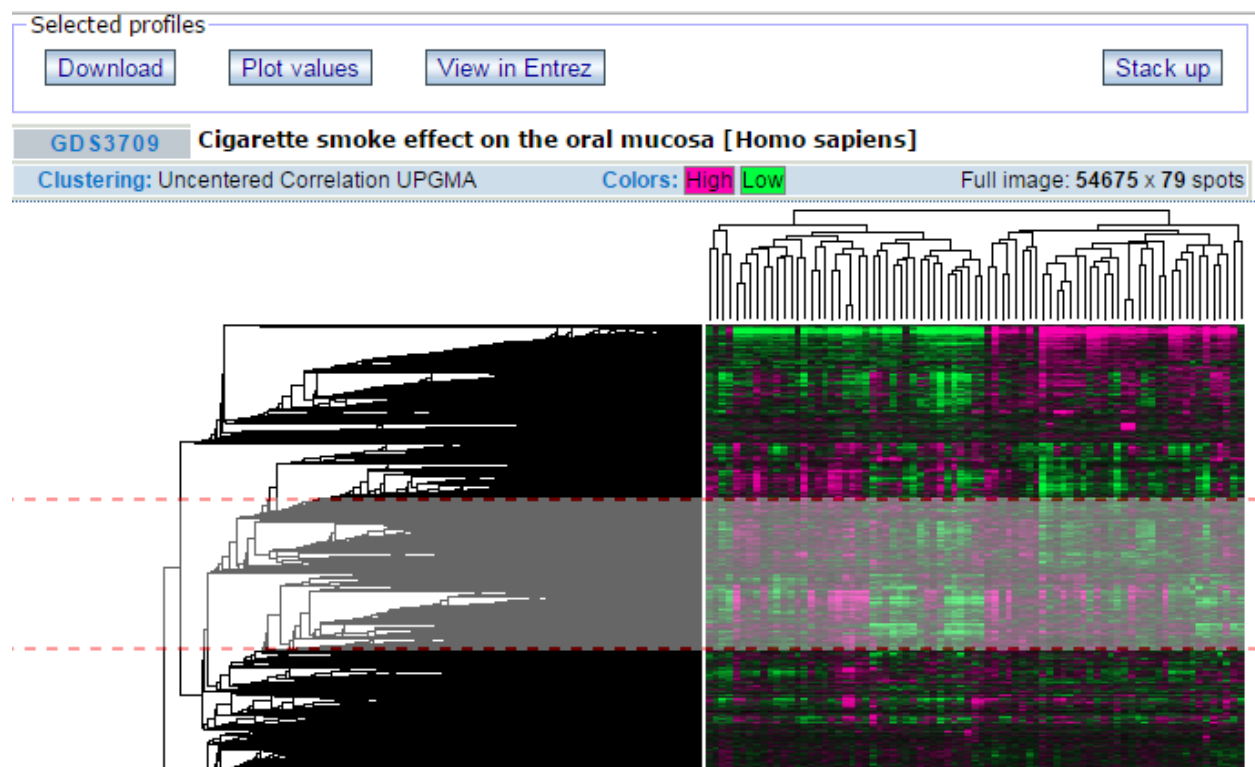


Figure 1.- Esquema de clustergrama realizado por GEO.

2.2 RStudio: Análisis de conjunto de datos

- Cree una carpeta, guarde su script y el conjunto de datos descargado.
- Para abrir los datos utilice el comando **getGEO** perteneciente a la librería **GEOquery**.

```
setwd("C:/Users/Manuel/Desktop/MI_PC/Doctorado/Bioinformática/Lab2015/BIOLAB4")
#source("http://www.bioconductor.org/biocLite.R")
#biocLite("Biobase")
#biocLite("GEOquery")
library(Biobase)
```

```
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##   clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##   clusterExport, clusterMap, parApply, parCapply, parLapply,
##   parLapplyLB, parRapply, parSapply, parSapplyLB
##
## The following object is masked from 'package:stats':
##
##   xtabs
##
## The following objects are masked from 'package:base':
##
##   anyDuplicated, append, as.data.frame, as.vector, cbind,
##   colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##   intersect, is.unsorted, lapply, Map, mapply, match, mget,
##   order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##   rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
##   table, tapply, union, unique, unlist, unsplit
##
## Welcome to Bioconductor
##
##   Vignettes contain introductory material; view with
##   'browseVignettes()'. To cite Bioconductor, see
##   'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
library(GEOquery)
```

```
## Setting options('download.file.method.GEOquery'='auto')
```

```
datos <- getGEO(filename='GDS3709.soft.gz')
```

- Averigue las características principales de su conjunto de datos utilizando los siguientes comandos.

```
Meta(datos)$channel_count
```

```
## [1] "1"
```

```
Meta(datos)$description
```

```
## [1] "Analysis of oral mucosae from 40 cigarette smokers and 40 age and gender matched never-smokers."
## [2] "female"
## [3] "male"
## [4] "cigarette smoke"
## [5] "control"
```

```
Meta(datos)$feature_count
```

```
## [1] "54675"
```

```
Meta(datos)$platform
```

```
## [1] "GPL570"
```

```
Meta(datos)$sample_count
```

```
## [1] "79"
```

```
Meta(datos)$sample_organism
```

```
## [1] "Homo sapiens"
```

```
Meta(datos)$sample_type
```

```
## [1] "RNA"
```

```
Meta(datos)$title
```

```
## [1] "Cigarette smoke effect on the oral mucosa"
```

```
Meta(datos)$type
```

```
## [1] "Expression profiling by array" "gender"  
## [3] "gender"                      "agent"  
## [5] "agent"
```

2.3 RStudio: conversión de datos a variables ordenadas.

- Para convertir los datos desde la estructura de Bioconductor a estructuras manejables directamente por R, se utiliza el comando **GDS2eset**

```
eset <- GDS2eSet(datos, do.log2=TRUE,GPL=NULL,AnnotGPL=FALSE)
```

```
## Warning in download.file(myurl, destfile, mode = mode, quiet = TRUE, method  
## = getOption("download.file.method.GEOquery")): downloaded length 65232231 !  
## = reported length 200
```

```
## File stored at:  
## C:\Users\Manuel\AppData\Local\Temp\RtmpU1cBlF/GPL570.soft
```

- Revise la estructura de los datos llamando a la variable anterior.

```
eset
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 54675 features, 79 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM447401 GSM447411 ... GSM447476 (79 total)
##   varLabels: sample gender agent description
##   varMetadata: labelDescription
## featureData
##   featureNames: 1007_s_at 1053_at ... AFFX-TrpnX-M_at (54675
##     total)
##   fvarLabels: ID GB_ACC ... Gene Ontology Molecular Function (16
##     total)
##   fvarMetadata: Column labelDescription
## experimentData: use 'experimentData(object)'
##   pubMedIds: 20179299
## Annotation:
```

- Conversión de datos a variables simples.

```
#-----
#Matriz de Expresion
#-----
mat_exp<-matrix(0, nrow = 54675, ncol = 79)
for(a in 1:54675)
  {for(b in 1:79)
    {mat_exp[a,b]<-exprs(eset)[a,b]
    }
  }
print(mat_exp[1:10,1:5])
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 10.134426 10.224002 9.940754 10.070121 10.121534
## [2,]  7.667466  7.503826 7.842979  7.325530  7.710806
## [3,]  5.738768  5.797013 5.615887  5.289097  5.698774
## [4,]  7.940167  7.710118 7.982423  7.802839  7.543032
## [5,]  3.040542  3.023433 3.168963  2.921436  3.067467
## [6,]  7.297375  7.226894 7.036723  6.996615  7.267723
## [7,]  4.907371  4.985500 5.033423  5.178316  5.022812
## [8,]  4.952799  4.955127 5.265662  5.067381  5.346957
## [9,]  6.573496  6.368070 5.659354  5.172327  6.267161
## [10,] 6.352264  5.645010 5.815832  4.040892  6.259837
```

```
#-----
#Probes
#-----
probes<-featureData(eset)$"Gene Symbol"[1:54675]
print(probes[1:10])
```

```
## [1] DDR1   RFC2    HSPA6   PAX8    GUCA1A  UBA7    THRA    PTPN21  CCL5    CYP2E1
## 21050 Levels: ADAM32 AFG3L1P AKD1 ALG10 ARMCX4 ATP6V1E2 BEST4 ... FAM86B1
```

```

#-----
#Muestras y clases
#-----
muestras<-0
clase<-0
for(a in 1:79)
{muestras[a]<-sampleNames(phenoData(eset))[a]
  #No fumadores
  #Mujer
  if ((phenoData(eset)$"agent"[a]=="control")&(phenoData(eset)$"gender"[a]=="female"))
    {clase[a]=0}
  #Hombre
  if ((phenoData(eset)$"agent"[a]=="control")&(phenoData(eset)$"gender"[a]=="male"))
    {clase[a]=1}
  #Fumadores
  #Mujer
  if ((phenoData(eset)$"agent"[a]=="cigarette smoke")&(phenoData(eset)$"gender"[a]=="female"))
    {clase[a]=2}
  #Hombre
  if ((phenoData(eset)$"agent"[a]=="cigarette smoke")&(phenoData(eset)$"gender"[a]=="male"))
    {clase[a]=3}
}
print(muestras)

```

```

## [1] "GSM447401" "GSM447411" "GSM447413" "GSM447415" "GSM447416"
## [6] "GSM447425" "GSM447430" "GSM447435" "GSM447440" "GSM447444"
## [11] "GSM447448" "GSM447449" "GSM447450" "GSM447452" "GSM447458"
## [16] "GSM447461" "GSM447464" "GSM447468" "GSM447472" "GSM447400"
## [21] "GSM447402" "GSM447403" "GSM447405" "GSM447418" "GSM447422"
## [26] "GSM447424" "GSM447427" "GSM447428" "GSM447429" "GSM447431"
## [31] "GSM447432" "GSM447434" "GSM447442" "GSM447451" "GSM447462"
## [36] "GSM447463" "GSM447467" "GSM447469" "GSM447473" "GSM447404"
## [41] "GSM447406" "GSM447407" "GSM447409" "GSM447412" "GSM447426"
## [46] "GSM447433" "GSM447439" "GSM447441" "GSM447443" "GSM447445"
## [51] "GSM447446" "GSM447453" "GSM447455" "GSM447456" "GSM447459"
## [56] "GSM447466" "GSM447470" "GSM447474" "GSM447475" "GSM447398"
## [61] "GSM447399" "GSM447408" "GSM447410" "GSM447414" "GSM447417"
## [66] "GSM447419" "GSM447420" "GSM447421" "GSM447423" "GSM447436"
## [71] "GSM447437" "GSM447438" "GSM447447" "GSM447454" "GSM447457"
## [76] "GSM447460" "GSM447465" "GSM447471" "GSM447476"

```

```
print(clase)
```

```

## [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1
## [71] 1 1 1 1 1 1 1 1 1

```

- Cálculo de p-value y ordenamiento de datos. Se estudiarán mujeres fumadoras y mujeres no fumadoras.

```

#Nivel de significancia
pvalue<-0
c0 <- which(clase==0)

```

```

c1 <- which(clase==2)
num_col<-length(c0)+length(c1)
mat_exp2<-matrix(0, nrow = 54675, ncol = num_col)
for(a in 1:54675)
{
  mat_exp2[a,]=c(mat_exp[a,c0],mat_exp[a,c1])
  pvalue[a]<-round(t.test(mat_exp[a,c0], mat_exp[a,c1])$p.value, digits = 5)
}
clase2<-c(clase[c0],clase[c1])
clase2

```

```

## [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2

```

```

#Ordenar matriz
mat_exp2<-mat_exp2[order(pvalue),]
probes<-probes[order(pvalue)]
pvalue<-pvalue[order(pvalue)]

```

2.4 RStudio: gráfico de clustergrama.

Con objetivo de evaluar la existencia de diferencia de expresión, se construirá un clustergrama.

- Creación de paleta de colores

```

library(gplots)

##
## Attaching package: 'gplots'
##
## The following object is masked from 'package:stats':
##
##      lowess

color.map <- function(clase2) { if (clase2=="0") "#FF0000" else "#0000FF"}
patientcolors <- unlist(lapply(clase2, color.map))
patientcolors

## [1] "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000"
## [8] "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000"
## [15] "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#FF0000" "#0000FF"
## [22] "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF"
## [29] "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF" "#0000FF"
## [36] "#0000FF" "#0000FF" "#0000FF" "#0000FF"

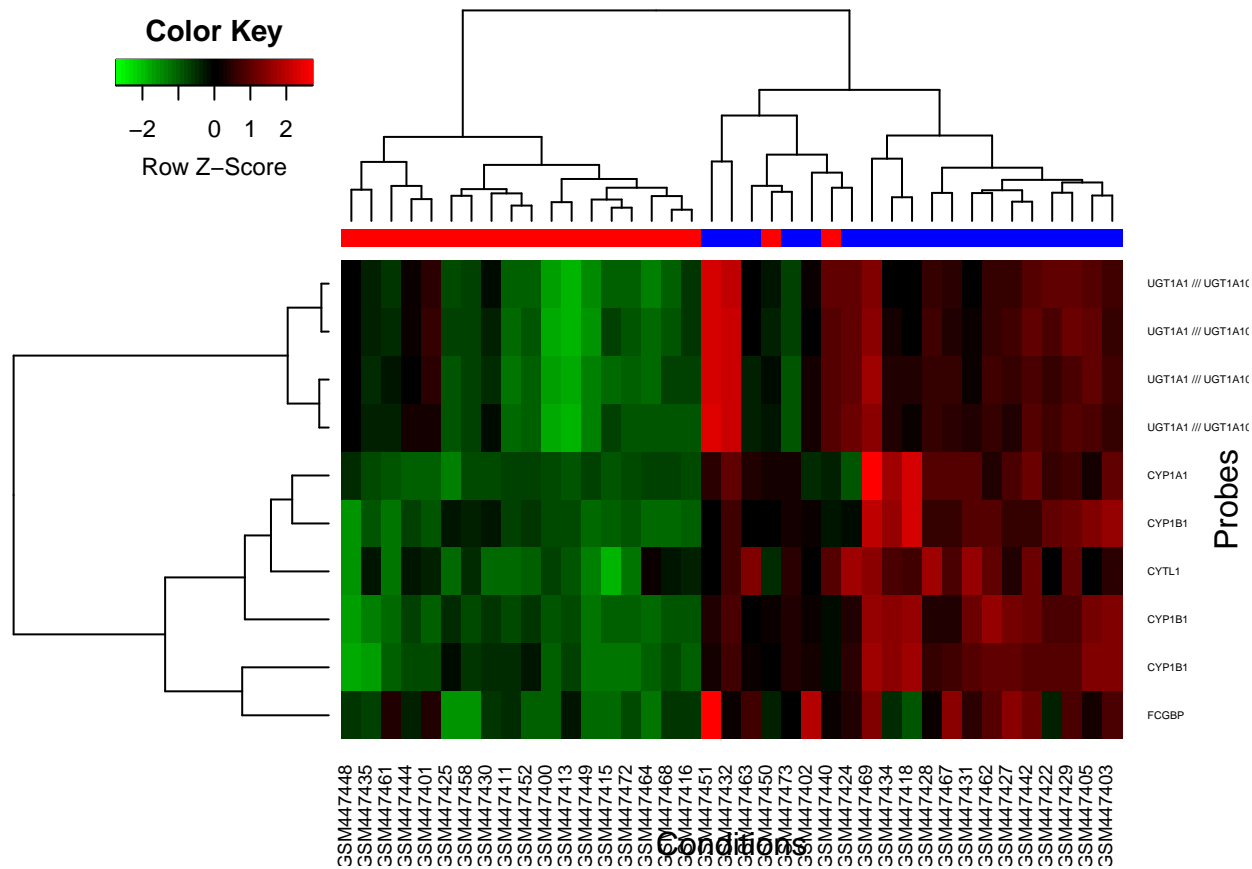
```

- Creación de clustergrama (heatmap)

```

heatmap.2(mat_exp2[1:10,],col=greenred(50),xlab='Conditions',ylab='Probes',
  labRow=probes[1:10],ColSideColors=patientcolors, scale="row", key=TRUE,
  symkey=FALSE, density.info="none", trace="none", cexRow=0.5,
  distfun = function(x) dist(x,method = 'euclidean'),labCol=muestras,
  hclustfun = function(x) hclust(x,method = 'complete'))

```

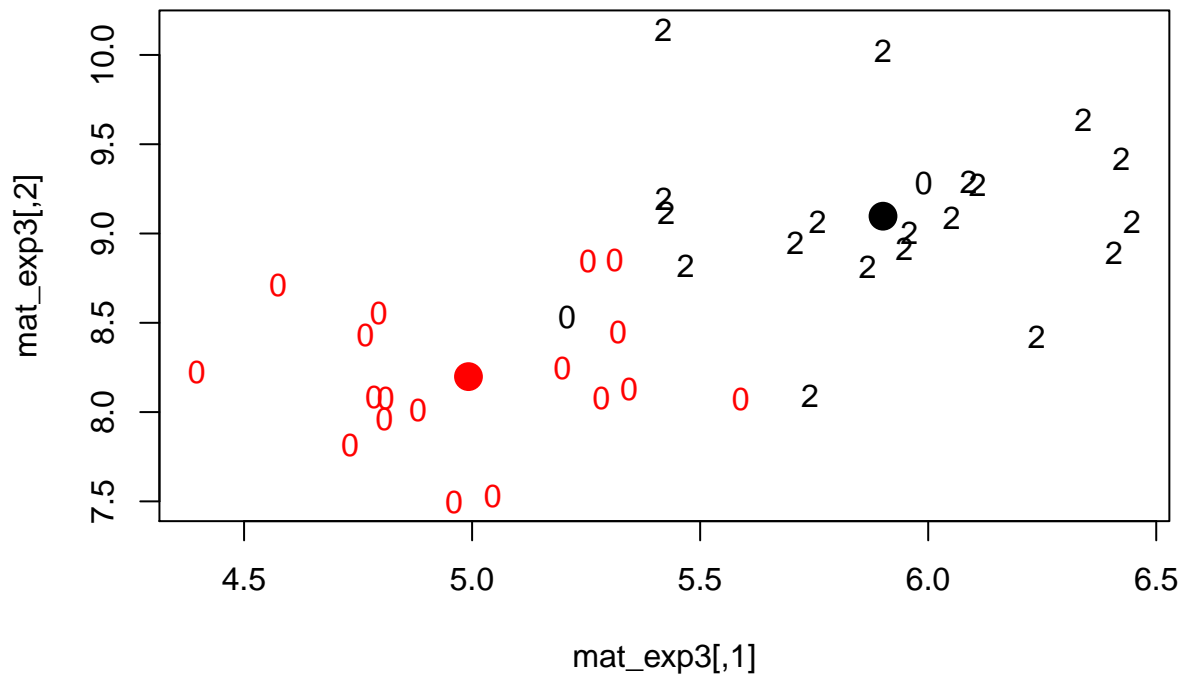



2.5 RStudio: K-means

- Aplicación de K-means para evaluar capacidad de discriminación.

```
mat_exp3 <- t(mat_exp2[1:10,])[,nrow(mat_exp2[1:10,]):1]
cl <- kmeans(mat_exp3, 2)
grafico<-plot(mat_exp3, col = cl$cluster, type='n')
text(mat_exp3, labels=clase2, col=cl$cluster)
points(cl$centers, col = 1:2, pch = 16, cex = 2)
title(main="Aplicación de K-means")
```

Aplicación de K-means



3. Actividades para desarrollar

- ¿Qué puede concluir de los resultados anteriores?
- ¿Qué etapas falta considerar en el proceso de análisis de expresión génica?
- ¿Se repetirán los mismos resultados para hombres fumadores y hombres no fumadores?
- ¿Se repetirán los mismos resultados entre fumadores y no fumadores?