Actividad 4

Definimos la opcion para que no se lean los strings como factores.

```
options(stringsAsFactors=F)
```

Carga del archivo

```
setwd("C:\\Users\\Choy\\Documents\\Semestre 2\\Análisis de biología computacional")
load("TCGA_COADREAD_comp_data.RData")
```

Qué contiene este Rdata Se cargan dos objetos tcga_coadread que contiene datos rnaseq (secuencias de RNA) con la expresion de genes y tcga coadread class con la clasificación de muestras entre old y young.

```
ls()
```

```
## [1] "tcga_coadread"       "tcga_coadread_class"
```

```
tcga_coadread[1:5,1:3]
```

```
##       TCGA-4N-A93T-01A-11R-A37K-07 TCGA-5M-AAT4-01A-11R-A41B-07
## A1BG                     8.4201199                     3.699813
## A1CF                     7.6529365                     8.242837
## A2BP1                    4.3482675                     1.892123
## A2LD1                    9.5366939                     7.994181
## A2ML1                    0.6732702                     3.790944
##       TCGA-5M-AAT6-01A-11R-A41B-07
## A1BG                      6.638268
## A1CF                      0.000000
## A2BP1                     2.746614
## A2LD1                     7.493938
## A2ML1                     2.441970
```

```
#data.frame(tcga_coadread)
```

```
tcga_coadread_class[1:10]
```

```
##  [1] "Old"   "Old"   "Young" "Old"   "Old"   "Old"   "Old"   "Young" "Old"
## [10] "Old"
```

```
as.data.frame(table(tcga_coadread_class))
```

```
##   tcga_coadread_class Freq
## 1                 Old  202
## 2               Young   84
```

Obtener genes diferencialmente expresados con una prueba t Student y la diferencia de medias

Calculamos la diferencia entre las medias tal como lo hicimos en la Actividad 2, primero obtendremos los índices para cada tipo de muestra y luego hacemos el cálculo

```
y_muestras<-which(tcga_coadread_class=="Young")
o_muestras<-which(tcga_coadread_class=="Old")
```

Definimos la matriz para el cálculo de diferencias entre medias

```
matriz_medias<- matrix(NA, nrow=nrow(tcga_coadread), ncol=3,
dimnames=list(rownames(tcga_coadread),c("Young","Old", "Diferencia")))
matriz_medias[1:5,]
```

```
##       Young Old Diferencia
## A1BG     NA  NA         NA
## A1CF     NA  NA         NA
## A2BP1    NA  NA         NA
## A2LD1    NA  NA         NA
## A2ML1    NA  NA         NA
```

La siguiente será la matriz que además de las diferencias, mostrará los p values y el FC.

```
matriz_ttest <- matrix(NA, nrow=nrow(tcga_coadread), ncol=4,
dimnames=list(rownames(tcga_coadread),c("Young","Old", "P value", "Fold change")))
matriz_ttest[1:5,]
```

```
##       Young Old P value Fold change
## A1BG     NA  NA      NA          NA
## A1CF     NA  NA      NA          NA
## A2BP1    NA  NA      NA          NA
## A2LD1    NA  NA      NA          NA
## A2ML1    NA  NA      NA          NA
```

Corremos un ciclo for para obtener los promedios y sus diferencias para cada gen.

```
for(i in 1:nrow(tcga_coadread)){
  media_young <- mean(tcga_coadread[i, y_muestras])
  media_old <- mean(tcga_coadread[i, o_muestras])
  aux_diferencia <- abs(media_young-media_old)
  matriz_medias[i,] <- c(media_young, media_old, aux_diferencia)
}
matriz_medias[1:5,]
```

```
##          Young      Old Diferencia
## A1BG  5.287141 5.222819 0.06432163
## A1CF  7.863854 7.933667 0.06981268
## A2BP1 2.128594 2.381282 0.25268822
## A2LD1 8.166356 8.148193 0.01816252
## A2ML1 1.023739 1.039845 0.01610567
```

Corremos un ciclo for para obtener los promedios, sus diferencias, el p value y el fold change para cada gen

```
for(i in 1:nrow(tcga_coadread)){
  media_young <- mean(tcga_coadread[i, y_muestras])
  media_old <- mean(tcga_coadread[i, o_muestras])
  p_value <- t.test(tcga_coadread[i, y_muestras], tcga_coadread[i, o_muestras])$p.value
  fold_change <- media_young -media_old
  matriz_ttest[i,] <- c(media_young, media_old,p_value, fold_change)
}
matriz_ttest[1:5,]
```

```
##          Young      Old   P value Fold change
## A1BG  5.287141 5.222819 0.6298451  0.06432163
## A1CF  7.863854 7.933667 0.7568771 -0.06981268
## A2BP1 2.128594 2.381282 0.4027582 -0.25268822
## A2LD1 8.166356 8.148193 0.8362485  0.01816252
## A2ML1 1.023739 1.039845 0.9446485 -0.01610567
```

```
#data.frame(matriz_ttest)
head(matriz_ttest)
```

```
##           Young       Old   P value Fold change
## A1BG   5.287141  5.222819 0.6298451  0.06432163
## A1CF   7.863854  7.933667 0.7568771 -0.06981268
## A2BP1  2.128594  2.381282 0.4027582 -0.25268822
## A2LD1  8.166356  8.148193 0.8362485  0.01816252
## A2ML1  1.023739  1.039845 0.9446485 -0.01610567
## A2M   13.331059 13.166404 0.2380401  0.16465505
```

Mayores diferencias

```
top_dif<-matriz_medias[order(-data.frame(matriz_medias)$Diferencia),]
#data.frame(top_dif)[1:50,]
head(top_dif)
```

```
##           Young      Old Diferencia
## GATA4  3.317011 1.599912   1.717099
## PCSK1N 5.168456 3.549272   1.619184
## PIWIL1 4.278384 5.787016   1.508632
## XIST   8.055190 6.685698   1.369492
## DUSP27 7.197303 5.877695   1.319608
## HAVCR1 4.642326 3.337936   1.304390
```

P values más pequeños, significativos.

```
top_p_value<-matriz_ttest[order(data.frame(matriz_ttest)$P.value),]
#data.frame(top_p_value)[1:50,]
head(top_p_value)
```

```
##               Young       Old      P value Fold change
## MTERF    8.39193346 7.6258149 8.054799e-07  0.76611852
## PRND     4.16365896 2.9448635 4.748019e-06  1.21879543
```

```
## FZD9     4.51446092 3.3978873 1.000739e-05  1.11657367
## MLF1     8.38633153 7.4555142 1.630553e-05  0.93081730
## TBC1D3P2 0.01050615 0.0882879 6.441823e-05 -0.07778175
## PCSK1N   5.16845587 3.5492721 7.661858e-05  1.61918372
```

Algunos genes tienen muy poca expresión (como el TBC1D3P2), así que los descriminaremos de nuestro análisis.

```
index_filter_exp <- which(apply(matriz_ttest[,1:2], 1, function(x) all(x < 1)))
matriz_medias <- matriz_medias[-index_filter_exp,]
matriz_ttest <- matriz_ttest[-index_filter_exp,]
```

¿Hay coincidencias entre aquellos genes con menor p value y aquellos genes con mayor diferencia de expresión? Haremos una revisión de los primeros 50 de cada lista.

```
coincidencias<-intersect(rownames(top_dif)[1:50],rownames(top_p_value)[1:50])
coincidencias
```

```
##  [1] "GATA4"  "PCSK1N" "PIWIL1" "HAVCR1" "DSC3"   "DKK1"   "PRND"   "FOLR1"
##  [9] "GAL"    "FZD9"   "BHMT"   "MLF1"   "SH3GL2" "RPL39L"
```

```
matriz_index<- matrix(NA, nrow=length(coincidencias), ncol=2,
dimnames=list(coincidencias,c("Índice Top Diferencias", "Índice menor P value")))
matriz_index[1:5,]
```

```
##        Índice Top Diferencias Índice menor P value
## GATA4                      NA                   NA
## PCSK1N                     NA                   NA
## PIWIL1                     NA                   NA
## HAVCR1                     NA                   NA
## DSC3                       NA                   NA
```

```
for(i in 1:length(coincidencias)){
  index_top_dif <- which(rownames(top_dif)==coincidencias[i])
  index_top_p_value <-which(rownames(top_p_value)==coincidencias[i])
  matriz_index[i,] <- c(index_top_dif, index_top_p_value)
}
matriz_index
```

```
##        Índice Top Diferencias Índice menor P value
## GATA4                       1                   17
## PCSK1N                      2                    6
## PIWIL1                      3                    8
## HAVCR1                      6                   36
## DSC3                        7                   41
## DKK1                        8                   25
## PRND                        9                    2
## FOLR1                      10                   47
## GAL                        12                   15
## FZD9                       14                    3
## BHMT                       20                   26
## MLF1                       31                    4
## SH3GL2                     44                   31
## RPL39L                     45                   43
```

De la tabla anterior parecería muy interesante poner atención a genes como el GATA4, el PCSK1N, el PIWIL1, el PRND O el FZD9, ya que la brecha entre su expresión en jóvenes y adultos mayores es grande y simultáneamente su p value es significativo. Sin embargo, también falta por comparar con el Fold change.

Haremos una selección más fina de genes, tomando a aquellos cuyo p value es menor a 0.01 (estrictamente significativos), y después los clasificaremos según si están sobreexpresados (positivos) o subexpresados (negativos) utilizando el Foldchange.

Definimos la matriz "matriz_ttest_pval" para tener los genes ordenados por p-value.

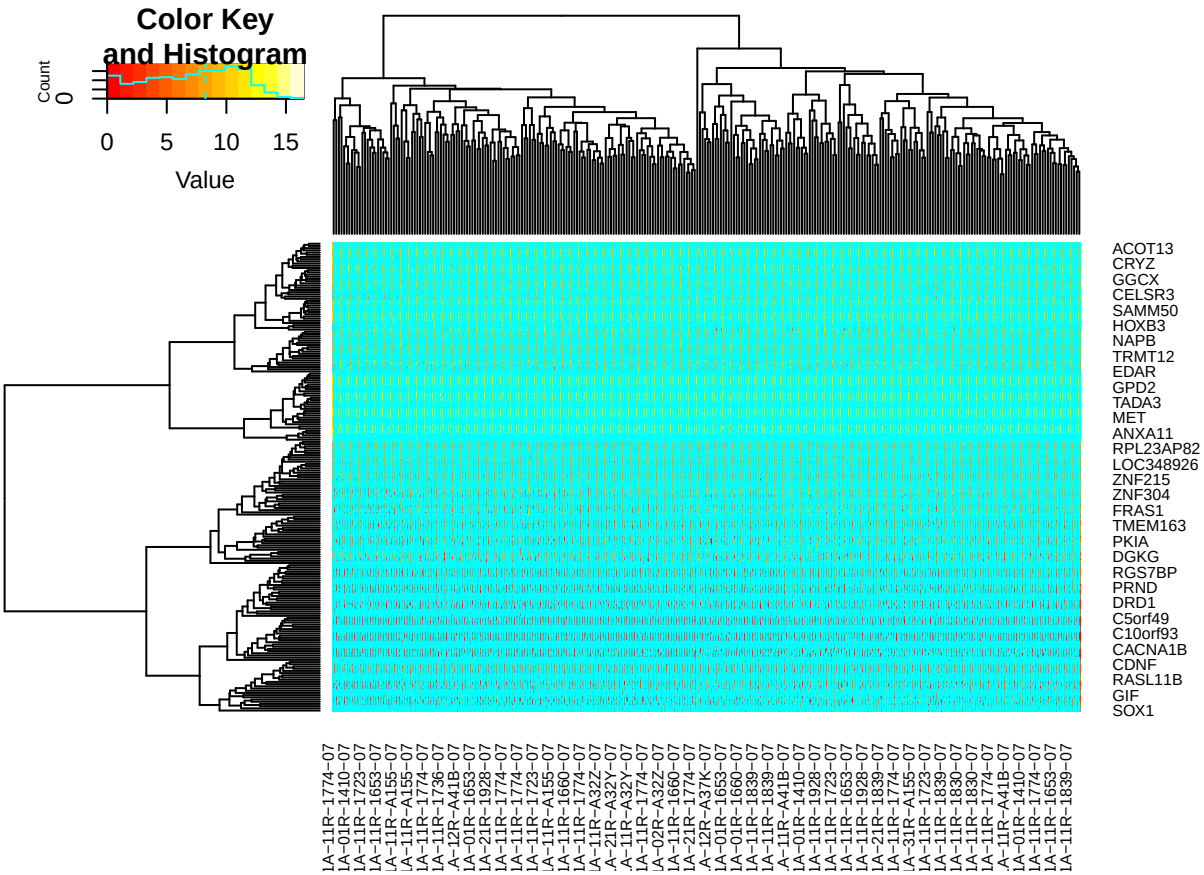
```
index_order_pvals <- order(abs(matriz_ttest[,"P value"]))

matriz_ttest_pval <- matriz_ttest[index_order_pvals,]

index_de_high <- which(matriz_ttest_pval[,"P value"] < 0.01 & matriz_ttest_pval[,"Fold change"] > 0)
de_genes_high <- rownames(matriz_ttest_pval)[index_de_high]

index_de_low <- which(matriz_ttest_pval[,"P value"] < 0.01 & matriz_ttest_pval[,"Fold change"] < 0)
de_genes_low <- rownames(matriz_ttest_pval)[index_de_low]
```

Al graficar un heatmap (mapa de calor) podemos ver el comportamiento de los genes diferencialmente expresados y las muestras por clase. Para hacer el heatmap necesitamos el paquete gplots. Guardamos en una matriz auxiliar los datos de los genes diferencialmente expresados (alto y bajo) ordenados por tipo de muestra.

```
#install.packages("gplots", dependencies=TRUE)
library("gplots") # load
```

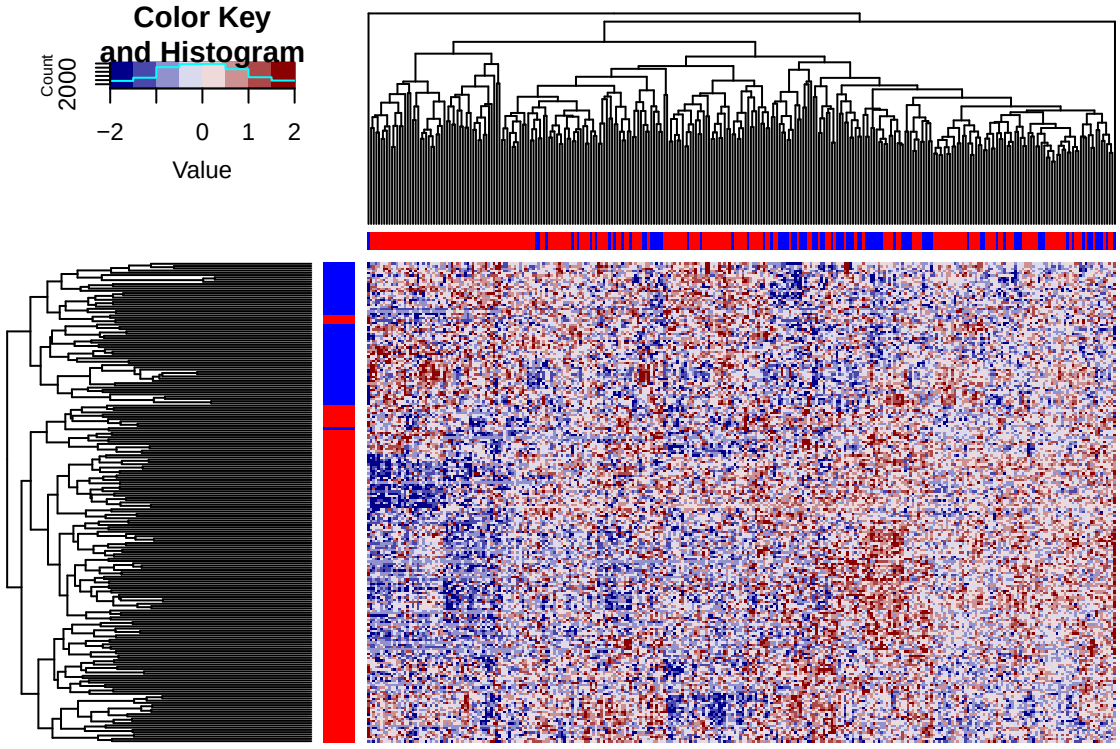
```
##
## Attaching package: 'gplots'

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

```
index_order_class <- order(tcga_coadread_class)
row_colors <- c(rep("red",length(de_genes_high)), rep("blue",length(de_genes_low)))
col_colors <- ifelse(tcga_coadread_class[index_order_class] == "Young","blue","red")
aux_mat <- tcga_coadread[c(de_genes_high, de_genes_low), index_order_class]
heatmap.2(aux_mat)
```



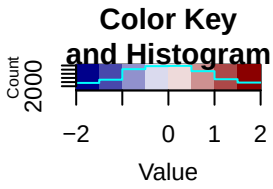
Para mejorar la visualización, escalamos los valores de expresión con la función scale, la cual hace una transformación z. Definimos el rango de colores y la cantidad de particiones de color (breaks) para tener un buen balance de valor de expresión y color. El heatmap se genera con la función heatmap.2. También agregamos colores para visualizar el tipo de muestra en las columnas (azul para joven, rojo para adulto mayor) y los genes diferencialmente expresados por renglón (rojo para alto y azul para bajo).

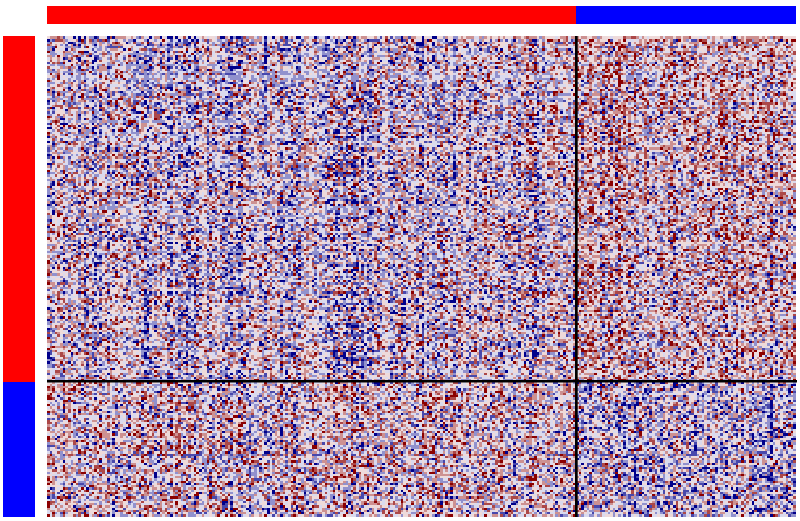
```
aux_mat <- t(apply(aux_mat, 1, scale))
colnames(aux_mat) <- colnames(tcga_coadread)
colors_h <- colorRampPalette(c("darkblue","white","darkred"))(8)
h_breaks <- seq(from=-2, to=2, length=9)
heatmap.2(aux_mat, col=colors_h, trace="none", breaks=h_breaks, labRow="", labCol="",
ColSideColors = col_colors, RowSideColors = row_colors)
```



Si observamos las ramas del agrupamiento jerárquico en columnas, se distingue que no hay una clara separación entre jóvenes y adultos mayores, ya que se forman pequeños grupos pero no están apartados. En cambio para los genes diferencialmente expresados en renglones, la separación es obvia. También podemos hacer el heatmap sin el cluster jerárquico para ver si detectamos patrones. Agregamos un separador de columnas y renglones para distinguir el tipo de muestra y genes diferencialmente expresados. De esta manera se distingue un poco más los patrones de expresión.

```
heatmap.2(aux_mat, col=colors_h, trace="none", breaks=h_breaks, labRow="", labCol="", dendrogram='none'
```





Podemos imprimir los genes diferencialmente expresados en pantalla o en un archivo de texto, con la función write.table, para hacer el análisis de funciones de los genes usando la herramienta MSigDB ¿tal como lo vimos en clase? En este caso imprimimos algunos en pantalla, pero se puede pasar a un archivo de texto especificando el parámetros "file".

Analizaremos con MSigDB los 153 genes de la lista de sobreexpresados.

```
length(de_genes_high)
```

```
## [1] 153
```

```
write.table(de_genes_high[1:153], sep="\t", quote=F, row.names=F, col.names=F)
```

```
## MTERF
## PRND
## FZD9
## MLF1
## PCSK1N
## LOC100009676
## KCNS3
## C5orf49
## CCDC90B
## TRMT12
## GAL
## SPATA17
## GATA4
```

```
## ZNF75A
## GAMT
## CCDC67
## DKK1
## BHMT
## MYL3
## TFAP2E
## TPPP3
## SH3GL2
## HLTF
## NAT8L
## ACOT13
## HAVCR1
## FRAS1
## CDH7
## HSCB
## DSC3
## TMSB15A
## RPL39L
## GLT8D1
## FOLR1
## ZMAT5
## SLC46A1
## GHDC
## GREB1L
## LOC348926
## SLC8A2
## TBCK
## C16orf71
## LEFTY2
## AHSG
## MAP9
## GLDC
## CBY1
## HSPC157
## MGA
## GDF11
## DNM3
## C1orf89
## TMEM184B
## GSTZ1
## PEG10
## CCDC102A
## C3orf14
## MARK1
## NHP2L1
## KCNG3
## HAAO
## DRD1
## CCDC103
## BLMH
## SPATA12
## CYB5D2
## ZNF879
```

```
## KLRG2
## GGCX
## BCDIN3D
## C22orf28
## C14orf105
## LRRC55
## CHFR
## CRYZ
## C10orf93
## CACNA1B
## CETP
## RDH5
## DACH2
## LASS4
## AKR7A2
## GPR64
## MS4A15
## UBIAD1
## ENY2
## CLDN20
## IGSF1
## RIBC1
## TSPAN10
## LRTOMT
## MLH1
## TMEM163
## TTC25
## CBS
## KRTAP3-3
## OGDHL
## C11orf54
## SULT1E1
## SUMF1
## LOC253039
## ZSCAN23
## C11orf24
## SLC4A8
## BNIP3
## NPW
## CPS1
## TGFBR2
## ITFG3
## C9orf131
## WDR66
## SLC34A3
## SAMM50
## GABRB2
## SLC38A4
## TADA3
## RPL23AP82
## AIF1L
## CNNM1
## ALG12
## PRSS23
```

```
## ZNF582
## ZNF215
## PNMA6A
## CDNF
## BCL2L13
## GNASAS
## RASL11B
## RGS7BP
## NIPSNAP1
## FGF12
## PKIA
## GIF
## CPAMD8
## SCN4A
## B3GNT1
## SGSM3
## ZNF132
## FBXL2
## PCDHGB5
## XRCC6
## POLDIP3
## ZNF304
## PRKACB
## MYCN
## EXT1
## B3GALT2
## GPAA1
## FBXL20
## C4orf31
## MRM1
## COCH
## NKX2-1
```

Lo obtenido más relevante es lo siguiente:

GSE29618_PRE_VS_DAY7_POST_TIV_FLU_VACC ACCINE_PDC_UP [200]

Genes regulados al alza en comparación de las células dendríticas plasmacitoides (pDC) de la vacunación contra la influenza TIV antes de la vacunación versus las del día 7 después de la vacunación.

GSE9988_LPS_VS_LOW_LPS_MONOCYTE_UP [188]

Genes regulados al alza en comparación de monocitos tratados con 5000 ng / ml de LPS (agonista de TLR4) versus aquellos tratados con 1 ng / ml de LPS (agonista de TLR4).

GSE360_L_MAJOR_VS_T_GONDII_DC_UP [197]

Genes sobre-regulados en comparación con las células dendríticas (DC) expuestas a L. major versus DC expuestas a T. gondii.

Posteriormente hacemos lo mismo con los subexpresados, que son 60.

```
length(de_genes_low)
```

```
## [1] 60
```

```r
write.table(de_genes_low[1:60], sep="\t", quote=F, row.names=F, col.names=F)
```

```
## PIWIL1
## KCNRG
## ZNF239
## ZNF600
## YOD1
## ATG16L1
## AVPI1
## ANKMY1
## UBE2MP1
## C5orf56
## TRIM26
## DUSP5
## STK39
## CELSR3
## PPP1R15B
## SOX9
## IFNG
## ELF3
## AGAP7
## BATF2
## FZD5
## ANXA11
## LIF
## IL15RA
## SLC25A28
## HOXB8
## ZFP36
## EDAR
## IRF1
## TAP2
## GPD2
## ATF3
## SOX1
## ZC3H11A
## YBX2
## LPGAT1
## NAPB
## HSPA1B
## FOXD4L1
## CNTD2
## NR4A2
## ZNF384
## CLK2P
## SF3B4
## MET
## LRIT2
## ZBTB7B
## SOCS1
## TOB1
## MAT1A
## PCDHB14
```

```
## CSF2
## LOC729467
## HOXB5
## HOXB3
## SLC24A6
## FOS
## DGKG
## KLHL35
## ZFP36L2
```

Y las coincidencias más grandes están en los siguientes sets de genes

GSE14000_UNSTIM_VS_4H_LPS_DC_TRANSLATE ATED_RNA_DN [197]

Genes regulados negativamente en comparación del ARNm unido al polisoma (traducido) antes y 4 h después de la estimulación con LPS (agonista de TLR4).

HALLMARK_TNFA_SIGNALING_VIA_NFKB [200]

Genes regulados por NF-kB en respuesta a TNF [GeneID = 7124].

GSE41978_ID2_KO_VS_BIM_KO_KLRG1_LOW_EF _EFFECTOR_CD8_TCELL_UP [199]

Genes sobrerregulados en KLRG1 bajo [GeneID = 10219] células efectoras T CD8 durante la infección: knockout ID2 [GeneID = 10219] versus BCL2L11 [GeneID = 10018] knockout.

Por último, si nuestra búsqueda de MsigDB la hacemos con tanto los genes sobreexpresados como los subexpresados, tenemos los siguientes set de genes:

GSE29618_PRE_VS_DAY7_POST_TIV_FLU_VACC ACCINE_PDC_UP [200]

Genes regulados al alza en comparación de las células dendríticas plasmacitoides (pDC) de la vacunación contra la influenza TIV antes de la vacunación versus las del día 7 después de la vacunación.

GSE42021_TCONV_PLN_VS_CD24HI_TCONV_THY THYMUS_UP [200]

Genes sobrerregulados en T conv: ganglios linfáticos periféricos versus tímico CD24 alto [GeneID = 100133941].

GSE22025_UNTREATED_VS_TGFB1_TREATED_CD _CD4_TCELL_UP [199]

Genes regulados al alza en células T CD4 [GeneID = 920]: sin tratar versus TGFB1 [GeneID = 7040].