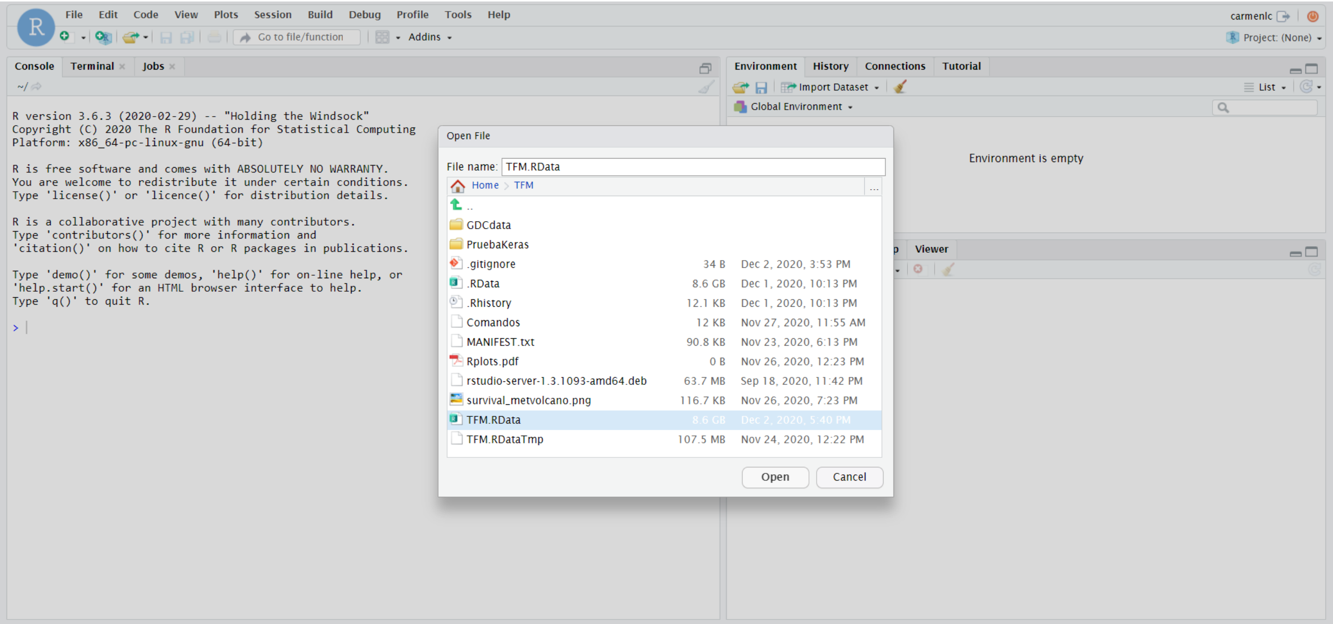
Informe 9: Creación modelos independientes en servidor UOC

Carmen Lebrero Cia

Contents

Abrir proyecto

Lo primero que tenemos que hacer es abrir nuestro proyecto de R que tenemos en el servidor `TFM.RData` mediante la aplicación que hemos habilitado de RStudio en el navegador de nuestro ordenador. Esto se puede conseguir desde la barra de tareas `File> Open File...` y se nos abrirá una ventana en pantalla para buscar nuestro archivo, que está dentro del directorio ‘`TFM` .



```
R version 3.6.3 (2020-02-29) -- "Holding the Windsock"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

File name: TFM.RData

```
..
GDCdata
PruebaKeras
.gitignore                            34 B   Dec 2, 2020, 3:53 PM
.RData                              8.6 GB   Dec 1, 2020, 10:13 PM
.Rhistory                          12.1 KB   Dec 1, 2020, 10:13 PM
Comandos                             12 KB   Nov 27, 2020, 11:55 AM
MANIFEST.txt                       90.8 KB   Nov 23, 2020, 6:13 PM
Rplots.pdf                             0 B   Nov 26, 2020, 12:23 PM
rstudio-server-1.3.1093-amd64.deb  63.7 MB   Sep 18, 2020, 11:42 PM
survival_metvolcano.png           116.7 KB   Nov 26, 2020, 7:23 PM
TFM.RData                           8.6 GB   Dec 2, 2020, 5:40 PM
TFM.RDataTmp                      107.5 MB   Nov 24, 2020, 12:22 PM
```

El objetivo de este informe es realizar modelos utilizando como información de entrada la de las ómicas por separado.

Puesta a punto de los datasets

Visualización de los archivos

Datos de Expresión génica con 238 genes (Tras análisis de expresión diferencial) ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed

```
> dim(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed)
[1] 238 290
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed)
 num [1:238, 1:290] 12.7 13.7 12 14.2 12 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:238] "ABCA1" "ACAT1" "ACO1" "ADAR" ...
  ..$ : chr [1:290] "TCGA-B4-5844-01A" "TCGA-B8-A54F-01A" "TCGA-BP-4760-01A" "TCGA-CJ-5675-01A" ...
```

Tenemos 238 genes (filas) y 290 columnas (muestas).

PCA

Expresión génica sin realizar análisis de expresión diferencial   ExpGenTCGA_KIRC_Norm_Trans_Filt75_

```
> dim(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed)
[1] 4897  290
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed)
 num [1:4897, 1:290] 17.5 12.1 12.8 12.7 12 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
  ..$ : chr [1:290] "TCGA-B4-5844-01A" "TCGA-B8-A54F-01A" "TCGA-BP-4760-01A" "TCGA-CJ-5675-01A" ...
```

Tenemos 4897 genes (filas) y 290 columnas (muestras).

Expresión proteica   ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renam

```
> dim(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)
[1] 177 290
> str(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)
'data.frame':   177 obs. of  290 variables:
 $ TCGA-B8-A54D-01A: num  0.065 -0.1759 -0.1956 -0.0203 -0.3174 ...
 $ TCGA-B0-5702-01A: num  0.08407 0.06711 -0.2797 -0.04449 0.00645 ...
 $ TCGA-CW-5588-01A: num  0.0859 -0.1164 0.1572 -0.3189 0.6729 ...
 $ TCGA-CJ-5682-01A: num  -0.0519 -0.0881 -0.2527 -0.2917 -0.6961 ...
 $ TCGA-CJ-4905-01A: num  0.1483 0.0691 -0.1798 -0.3004 -0.4388 ...
 $ TCGA-AK-3434-01A: num  0.0371 0.1184 0.4312 0.3344 -0.753 ...
 $ TCGA-B8-5164-01A: num  0.0852 -0.0261 -0.6137 0.306 -0.7781 ...
 $ TCGA-CZ-5985-01A: num  -0.0851 0.0103 0.4438 -0.2096 -0.3524 ...
 $ TCGA-BP-4993-01A: num  0.3912 0.3813 -0.0716 -0.0703 -0.9355 ...
 $ TCGA-CZ-4863-01A: num  0.01816 -0.00451 -0.06061 0.55619 -0.17628 ...
 $ TCGA-AK-3428-01A: num  0.441 0.471 0.904 -0.515 -0.752 ...
 $ TCGA-CJ-5679-01A: num  0.2566 -0.0766 -0.3716 0.2812 1.2605 ...
 $ TCGA-B0-4688-01A: num  0.0524 0.0208 0.3476 -0.6293 -1.4102 ...
 $ TCGA-B0-4694-01A: num  0.1601 0.0677 -0.0541 0.2397 0.5261 ...
 $ TCGA-B0-4701-01A: num  0.1785 0.1304 0.0697 -0.1694 -0.4759 ...
 $ TCGA-B8-4622-01A: num  0.151 0.017 0.279 -0.27 0.333 ...
 $ TCGA-G6-A8L8-01A: num  -0.0185 -0.1157 0.1092 0.2105 -0.3779 ...
```

Tenemos 177 proteínas (columnas) y 290 filas (muestras).

Metilación

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed)
[1] 373382    290
```

Tenemos 373382 sondas (filas) y 290 muestras (columnas).

Preparación de las etiquetas

Expresión génica genes = 238 ¿Cómo sacamos las etiquetas? Necesitamos obtener los ExpGenTCGA_KIRC_RawData$vital de los colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed) que sean iguales que ExpGenTCGA_KIRC_RawData$samples.

```
> x <- c()
> for (i in colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed)){
+ x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
+ }
> x
  [1]   5   7  10  14  17  20  21  22  25  26  27  29  30  31  32  35  36  37  40  44  47  50  52  53
 [25]  55  56  57  59  60  64  67  70  71  72  73  75  76  78  81  82  83  92  93  97  98 101 103 104
 [49] 109 110 112 113 114 116 118 120 121 122 123 124 127 129 130 132 133 135 136 137 139 143 146 147
 [73] 149 150 154 156 159 161 162 163 164 167 168 173 176 178 182 185 187 190 193 195 196 197 198 200
 [97] 201 202 205 208 209 211 216 218 219 224 227 228 229 230 231 233 234 235 236 237 238 240 244 245
[121] 250 254 255 256 257 261 263 264 265 267 268 273 275 277 280 281 283 293 295 296 299 302 307 308
[145] 311 312 313 314 315 319 322 324 325 326 330 331 332 335 336 337 339 340 343 345 346 349 350 351
[169] 353 356 357 358 360 362 363 364 365 366 367 369 370 372 378 383 384 385 386 387 388 389 390 394
[193] 395 396 397 398 402 405 408 409 417 421 422 423 427 432 435 436 437 440 442 443 444 445 446 448
[217] 449 450 451 455 456 458 459 460 463 464 465 468 469 470 471 476 477 478 482 483 484 486 487 488
[241] 490 491 492 495 496 502 503 504 505 507 509 510 511 515 517 519 520 525 527 528 531 533 538 539
[265] 540 542 543 546 549 551 555 560 564 567 571 572 573 574 577 581 582 587 593 597 598 599 600 602
[289] 603 605
> ExpGenTCGA_KIRC_RawData$sample[5]
[1] "TCGA-B4-5844-01A"
> colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed)[1]
[1] "TCGA-B4-5844-01A"
> ExpGenTCGA_KIRC_RawData$sample[7]
[1] "TCGA-B8-A54F-01A"
> colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed)[2]
[1] "TCGA-B8-A54F-01A"
```

En el vector x tenemos los índices de las muestras que coinciden entre ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSa y ExpGenTCGA_KIRC_RawData$sample. Es decir, la muestra que estaba en el índice 1 del vector colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed) se encuentra en la posición 5 del vector ExpGenTCGA_KIRC_RawData$sample.

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels <- ExpGenTCGA_KIRC_
```

Expresión génica genes = 4897 Necesitamos obtener los ExpGenTCGA_KIRC_RawData$vital_status de los colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed) que sean iguales que ExpGenTCGA_KIRC_RawData$samples.

```
> x <- c()
> for (i in colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed)){
+ x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
+ }
> x
 [1]   5   7  10  14  17  20  21  22  25  26  27  29  30  31  32  35  36  37  40  44  47  50  52  53
 [25]  55  56  57  59  60  64  67  70  71  72  73  75  76  78  81  82  83  92  93  97  98 101 103 104
 [49] 109 110 112 113 114 116 118 120 121 122 123 124 127 129 130 132 133 135 136 137 139 143 146 147
 [73] 149 150 154 156 159 161 162 163 164 167 168 173 176 178 182 185 187 190 193 195 196 197 198 200
 [97] 201 202 205 208 209 211 216 218 219 224 227 228 229 230 231 233 234 235 236 237 238 240 244 245
[121] 250 254 255 256 257 261 263 264 265 267 268 273 275 277 280 281 283 293 295 296 299 302 307 308
[145] 311 312 313 314 315 319 322 324 325 326 330 331 332 335 336 337 339 340 343 345 346 349 350 351
[169] 353 356 357 358 360 362 363 364 365 366 367 369 370 372 378 383 384 385 386 387 388 389 390 394
[193] 395 396 397 398 402 405 408 409 417 421 422 423 427 432 435 436 437 440 442 443 444 445 446 448
[217] 449 450 451 455 456 458 459 460 463 464 465 468 469 470 471 476 477 478 482 483 484 486 487 488
[241] 490 491 492 495 496 502 503 504 505 507 509 510 511 515 517 519 520 525 527 528 531 533 538 539
[265] 540 542 543 546 549 551 555 560 564 567 571 572 573 574 577 581 582 587 593 597 598 599 600 602
[289] 603 605
> ExpGenTCGA_KIRC_RawData$sample[5]
[1] "TCGA-B4-5844-01A"
> colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed)[1]
[1] "TCGA-B4-5844-01A"
```

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels <- ExpGenTCGA_KIRC_RawData
```

Expresión proteica Necesitamos obtener los `ExpGenTCGA_KIRC_RawData$vital_status` de los `colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)` que sean iguales que `ExpGenTCGA_KIRC_RawData$samples`.

```
> x <- c()
> for (i in colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)){
+ x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
+ }
> x
  [1] 114 595  52 593 504 232 254 187 582  37 490  69 211 153 449 601 465 238  97 600 149 242 139 440
 [25] 136 531 420 255 220 375 495 356 360 540 269 162  15 484  63 165 441 509 366 602  91 146 280 285
 [49] 557  65 566  53 460  30 314 486 330 454 487 389 362 323 173 433  14  55 286 237 428 129 547 251
 [73] 379 367  22 290 432  42  64 244 385 333   7 369 553 427 206 163  90 303 472 378  67 497  80 402
 [97] 599 451 434 339  96 358 277  19 488 166 555 372 581 156 545 514 523 233 312 344  78   8 305 506
[121] 343 185  21 147 240 597 519  23 386 143  29 528 123 234 565 307 403 257 405 336 348  28 461 189
[145] 563 580 477 444 560 443 325 583 168 567 483 164  73 517 421 522 491 396 535 293 349 321 300 245
[169] 525 130 198 133 205  36  48 157 377  72 500 289 150 345 320 101 549 265 302 110   1 281 414 529
[193]  84  92 452  83 207 527 117 505 492  40  93 438  86 283 572  13 331 425  77 250 365 408 180  26
[217] 411 196 328 182 370 564 218  76 160 158 347 335  57 319 571  10  44   4 532 437 496 550  82 503
[241] 102 100 120 383 542 466 337 493  75 186  47 439 455 276 464  66 387 548  99  50 353 231  74 148
[265] 259 423 352 179 229 273 124 436 327 407  20  35  32 538   5 268 401 175 274 406 368 216 132 272
[289]  31 275
> length(x)
[1] 290
> ExpGenTCGA_KIRC_RawData$sample[114]
[1] "TCGA-B8-A54D-01A"
> colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)[1]
[1] "TCGA-B8-A54D-01A"
```

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels <- ExpGenTCGA_KIRC_RawData$vita
```

Metilación

```
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels <- MetTCGA_KIRC_RawDa
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels
```

```
  [1] "Alive" "Alive" "Dead"  "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Alive"
 [13] "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Dead"  "Dead"  "Dead"  "Alive" "Alive" "Alive"
 [25] "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive"
 [37] "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive"
 [49] "Dead"  "Alive" "Dead"  "Dead"  "Alive" "Alive" "Alive" "Alive" "Dead"  "Dead"  "Alive" "Alive"
 [61] "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Dead"  "Dead"  "Alive" "Dead"  "Dead"
 [73] "Alive" "Dead"  "Alive" "Alive" "Alive" "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Alive" "Alive"
 [85] "Alive" "Alive" "Alive" "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive"
 [97] "Dead"  "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive" "Dead"
[109] "Dead"  "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Alive"
[121] "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Dead"  "Dead"  "Alive" "Alive" "Dead"
[133] "Dead"  "Dead"  "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"  "Alive" "Dead"  "Alive"
[145] "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Dead"  "Dead"  "Alive" "Alive"
[157] "Alive" "Dead"  "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"  "Alive" "Dead"  "Dead"
[169] "Alive" "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"  "Alive" "Alive" "Dead"
[181] "Alive" "Alive" "Dead"  "Alive" "Dead"  "Dead"  "Alive" "Alive" "Alive" "Alive" "Dead"  "Dead"
[193] "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Alive" "Alive" "Dead"  "Dead"  "Alive" "Alive" "Dead"
[205] "Alive" "Alive" "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Dead"  "Dead"  "Alive" "Dead"  "Alive"
[217] "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive"
[229] "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"  "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive"
[241] "Alive" "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Alive" "Alive" "Alive" "Dead"  "Dead"  "Alive"
[253] "Alive" "Alive" "Alive" "Dead"  "Dead"  "Dead"  "Dead"  "Alive" "Alive" "Dead"  "Dead"  "Alive"
[265] "Dead"  "Alive" "Dead"  "Alive" "Alive" "Alive" "Alive" "Alive" "Dead"  "Alive" "Alive" "Dead"
[277] "Alive" "Alive" "Dead"  "Alive" "Dead"  "Dead"  "Alive" "Alive" "Alive" "Dead"  "Alive" "Dead"
[289] "Alive" "Alive"
```

Creación de conjuntos de test, train y sus etiquetas correspondientes

Expresión génica genes = 238

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed <- t(ExpGenTCGA_
```

Creación set train y test para ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_S

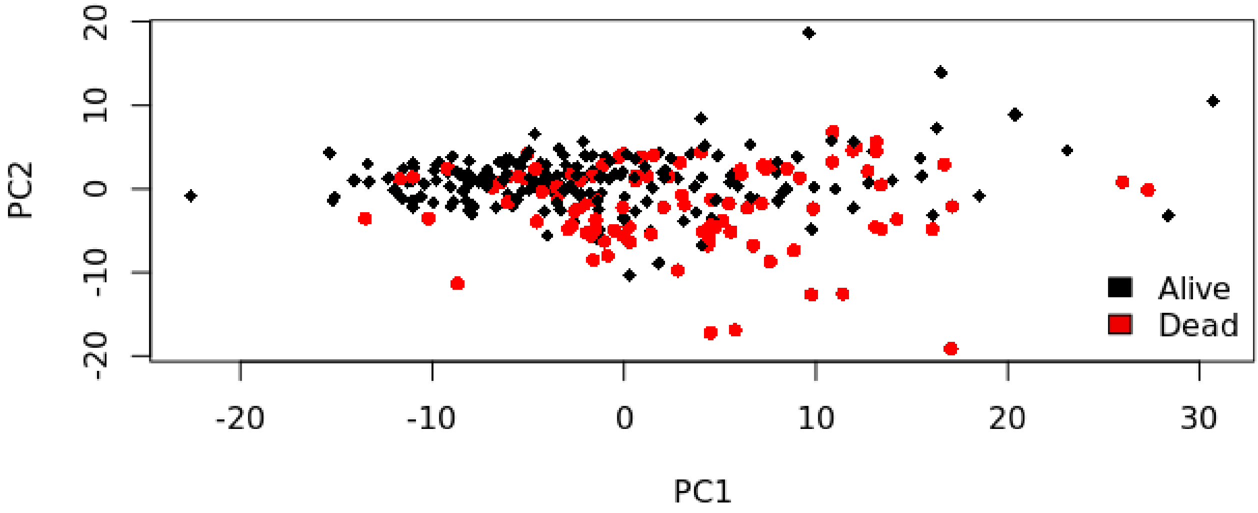
```
set.seed(231)
ExpGenTCGA_KIRC_Index_Training <- sample(1:nrow(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpPro
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Test <- ExpGenT
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Train <- ExpGen
```

Obtener etiquetas de cada set para ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpP

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test <- ExpGenTCGA_
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train <- ExpGenTCGA_

ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb <- a
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb[Exp

ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb <-
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb[Ex
```

PCA

```
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA <- prcomp
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA)
List of 5
 $ sdev    : num [1:238] 8.31 4.23 4.1 3.16 2.77 ...
 $ rotation: num [1:238, 1:238] -0.0724 -0.0863 -0.0722 -0.0727 -0.0796 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:238] "ABCA1" "ACAT1" "ACO1" "ADAR" ...
  .. ..$ : chr [1:238] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:238] 12.5 12.5 12 13.9 12.1 ...
  ..- attr(*, "names")= chr [1:238] "ABCA1" "ACAT1" "ACO1" "ADAR" ...
 $ scale   : logi FALSE
 $ x       : num [1:290, 1:238] -3.03 15.43 -3.49 -2.64 16.28 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:290] "TCGA-B4-5844-01A" "TCGA-B8-A54F-01A" "TCGA-BP-4760-01A" "TCGA-CJ-5675-01A" ...
  .. ..$ : chr [1:238] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels)
> c(18, 16)[type]
> plot(
+     ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Expresión génica genes = 4897

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed <- t(ExpGenTCGA_KIRC_N
)
```

Creación set train y test para ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSamp

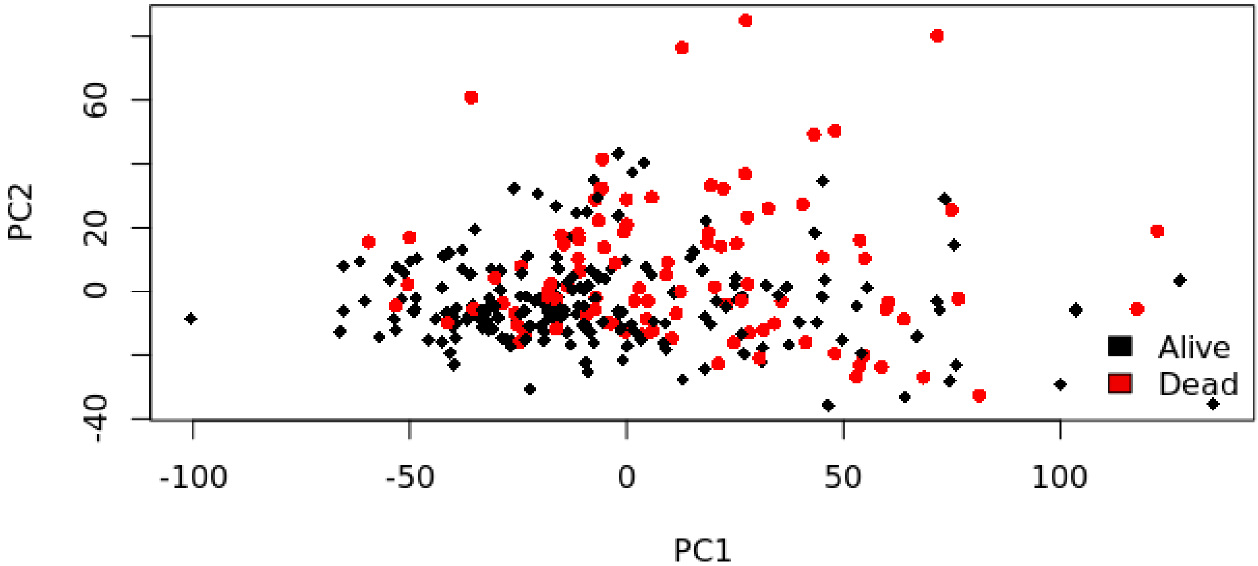
```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test <- ExpGenTCGA_KIRC
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train <- ExpGenTCGA_KI
```

Obtener etiquetas de cada set para ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Same

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test <- ExpGenTCGA_KIRC_No
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train <- ExpGenTCGA_KIRC_N

ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb <- as.inte

ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb[ExpGenTCGA_

ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb <- as.inte

ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb[ExpGenTCGA
```

PCA

```
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA <- prcomp(ExpGen
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA)
List of 5
 $ sdev    : num [1:290] 37.4 18.1 17.5 15.2 13 ...
 $ rotation: num [1:4897, 1:290] -0.0219 -0.0135 -0.012 -0.0165 -0.0132 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 16.3 12.3 12.7 12.5 11.9 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
 $ scale   : logi FALSE
 $ x       : num [1:290, 1:290] -17.24 71.64 -7.75 -13.03 74.41 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:290] "TCGA-B4-5844-01A" "TCGA-B8-A54F-01A" "TCGA-BP-4760-01A" "TCGA-CJ-5675-01A" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels)
> c(18, 16)[type]
> plot(
+     ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Expresión proteica

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed <- t(ExpProtTCGA_KIRC_RawDat
```

Creación set train y test para ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Test <- ExpProtTCGA_KIRC_Raw
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Train <- ExpProtTCGA_KIRC_Ra
```

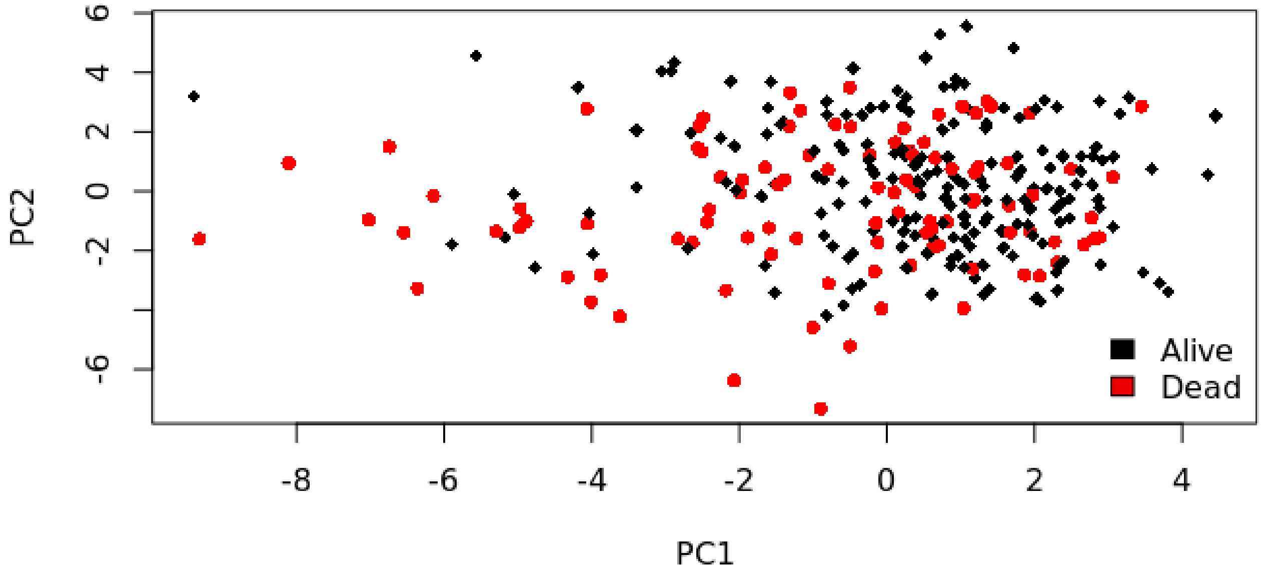
Obtener etiquetas de cada set para ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSam

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test <- ExpProtTCGA_KIRC_RawData
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train <- ExpProtTCGA_KIRC_RawDat

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb <- as.integer(f

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb[ExpProtTCGA_KIR

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb <- as.integer(

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb[ExpProtTCGA_KI
```

PCA

```
> ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_PCA <- prcomp(ExpProtTCGA_
> str(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_PCA)
List of 5
```

```
 $ sdev    : num [1:177] 2.36 2.21 1.8 1.47 1.32 ...
 $ rotation: num [1:177, 1:177] -0.045 -0.0308 -0.0373 0.0341 0.0209 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:177] "14-3-3_beta-R-V" "14-3-3_epsilon-M-C" "14-3-3_zeta-R-V" "4E-BP1_pS65-R-V" ...
  .. ..$ : chr [1:177] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:177] 0.0623 0.0081 0.0244 0.0185 0.0869 ...
  ..- attr(*, "names")= chr [1:177] "14-3-3_beta-R-V" "14-3-3_epsilon-M-C" "14-3-3_zeta-R-V" "4E-BP1_pS
 $ scale   : logi FALSE
 $ x       : num [1:290, 1:177] 0.259 -0.804 1.322 0.102 1.087 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:290] "TCGA-B8-A54D-01A" "TCGA-B0-5702-01A" "TCGA-CW-5588-01A" "TCGA-CJ-5682-01A" ...
  .. ..$ : chr [1:177] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels)
> c(18, 16)[type]
> plot(
+     ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Metilación

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed <- t(assay(MetTCGA_
> dim(t(assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed)))
[1]    290 373382
```

Creación set train y test para MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameS

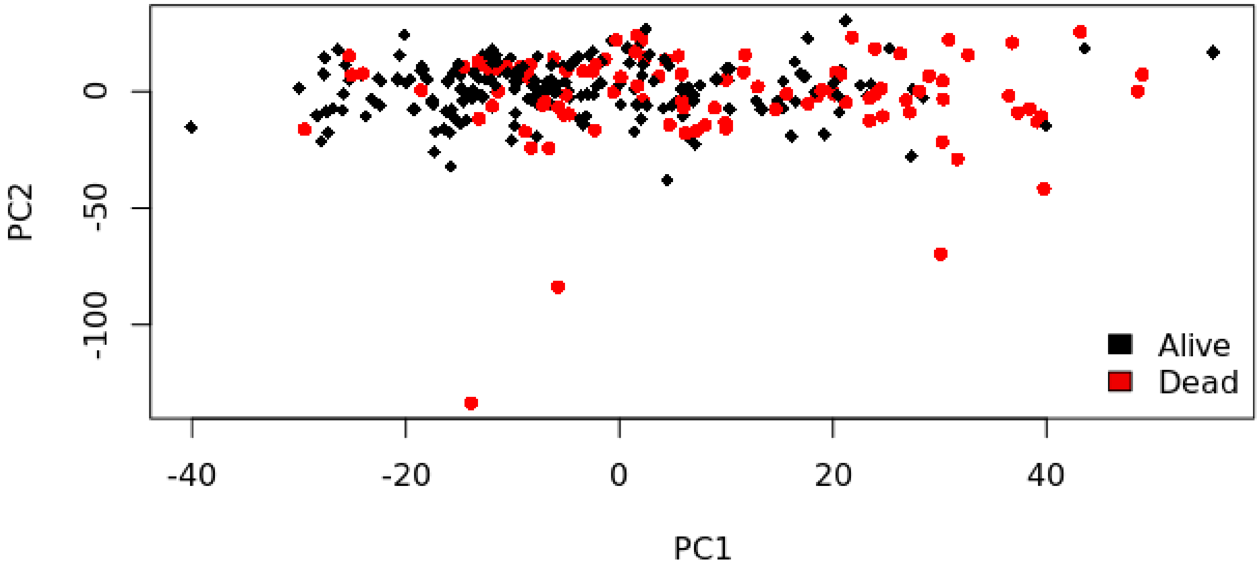
```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Test <- MetTCGA_KIR
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Train <- MetTCGA_KI
```

Obtener etiquetas de cada set para MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_S

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test <- MetTCGA_KIRC_Ra
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train <- MetTCGA_KIRC_R

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test_FactorNumb <- as.n

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test_FactorNumb[MetTCGA

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train_FactorNumb <- as

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train_FactorNumb[MetTCG
```

PCA

```
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_PCA <- prcomp(Met
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_PCA)
List of 5
 $ sdev    : num [1:606] 36.9 23.9 17.4 15.8 13.2 ...
 $ rotation: num [1:4897, 1:606] 0.0201 0.0136 0.013 0.0159 0.0117 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
  .. ..$ : chr [1:606] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 16.2 12.3 12.8 12.3 11.9 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
 $ scale   : logi FALSE
 $ x       : num [1:606, 1:606] -32.54 -7.53 -5.93 -1.55 17.1 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:606] "TCGA-B0-5694-01A-11R-1541-07" "TCGA-CJ-4637-01A-02R-1325-07" "TCGA-CZ-4860-01A-0
  .. ..$ : chr [1:606] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels)
> c(18, 16)[type]
> plot(
+     MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```

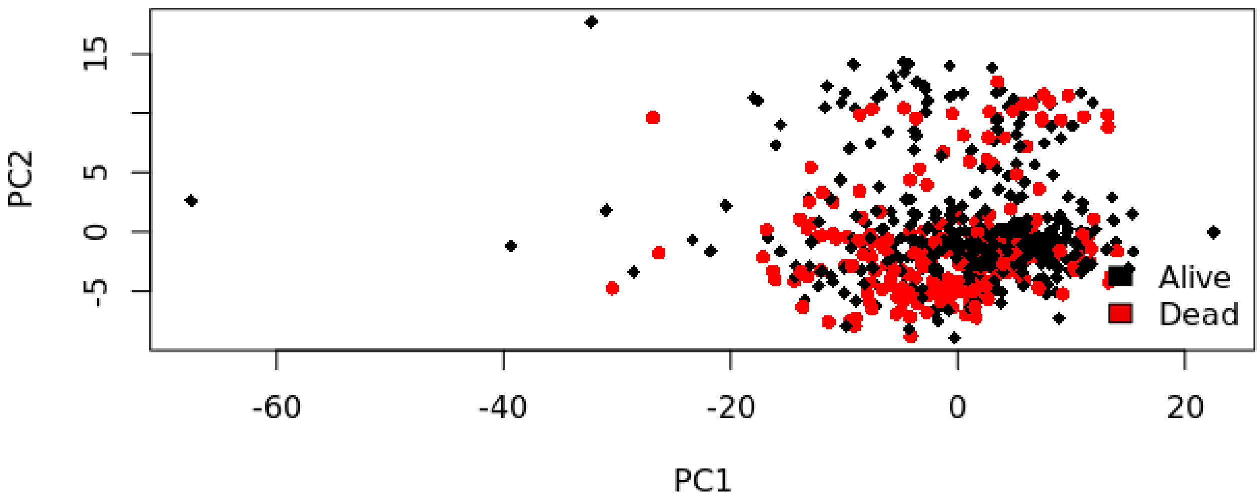


PCA de los datos sin haber unido los datos

Expresión génica (genes = 238)

```
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_Transposed <- t(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG)
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_Transposed_PCA <- prcomp(ExpGenTCGA_KIRC_Norm_Trans_Filt75_2
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_Transposed_PCA)
List of 5
 $ sdev    : num [1:238] 8.22 4.92 3.8 3.68 2.89 ...
 $ rotation: num [1:238, 1:238] 0.0657 0.0846 0.0735 0.0735 0.0836 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:238] "ABCA1" "ACAT1" "ACO1" "ADAR" ...
  .. ..$ : chr [1:238] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:238] 12.3 12.8 12.1 13.9 12.1 ...
  ..- attr(*, "names")= chr [1:238] "ABCA1" "ACAT1" "ACO1" "ADAR" ...
 $ scale   : logi FALSE
 $ x       : num [1:606, 1:238] -7.757 -2.067 -2.674 -0.556 2.908 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:606] "TCGA-B0-5694-01A-11R-1541-07" "TCGA-CJ-4637-01A-02R-1325-07" "TCGA-CZ-4860-01A-
  .. ..$ : chr [1:238] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_RawData$vital_status)
> c(18, 16)[type]
  [1] 16 16 16 16 18 18 18 18 18 18 18 16 16 18 18 18 18 18 16 18 18 18 16 16 18 18 18 18 18 18 16 18
 [33] 16 18 18 16 18 16 16 18 18 16 18 18 18 16 18 16 18 16 16 18 18 18 16 18 18 18 18 18 18 18 18 18
 [65] 18 16 18 18 16 18 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 18 18 16 16 18 18 18 18 18 18 18
 [97] 16 18 18 18 18 18 16 18 16 16 16 18 18 16 16 18 16 18 16 18 18 16 16 18 18 18 16 16 18 18 16 16
[129] 18 16 18 18 18 18 18 18 18 18 18 18 16 18 18 18 18 18 18 18 18 18 18 18 16 16 16 16 16 18 16 16
```

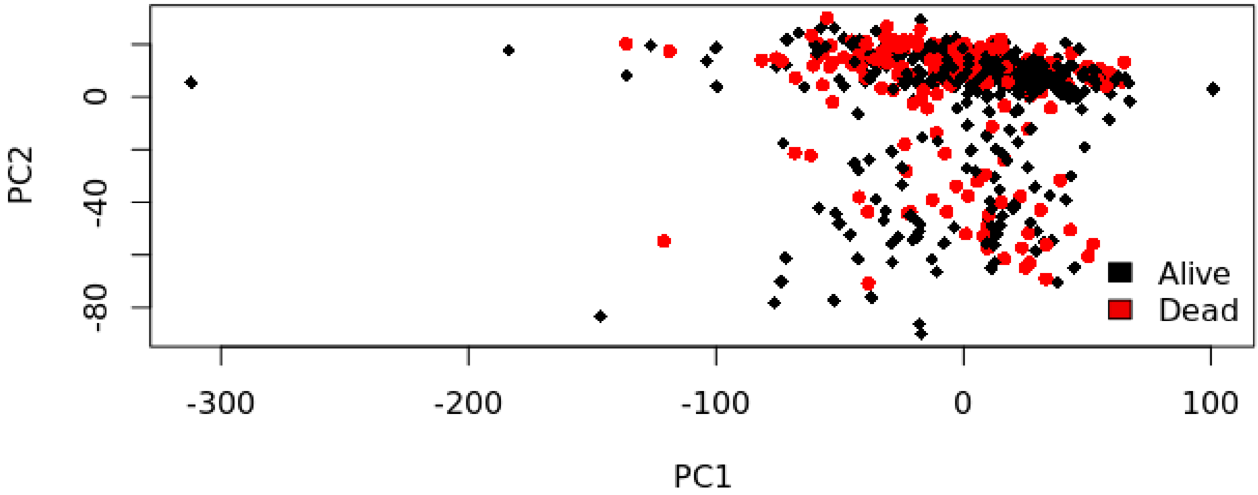
```
[161] 18 18 16 16 18 18 16 18 16 18 16 16 18 16 18 16 16 18 18 18 18 18 16 16 16 18 18 18 18 18 18 18
[193] 16 18 18 18 18 18 18 16 16 18 18 18 16 16 18 16 18 18 16 18 18 18 18 18 18 18 16 16 18 16 18 16
[225] 18 18 18 18 16 18 18 18 18 16 16 18 18 18 18 18 18 18 16 18 16 18 18 18 16 18 16 18 18 16 18 18
[257] 16 18 16 18 18 16 18 16 16 18 16 16 18 18 18 18 18 16 18 18 16 16 18 18 18 18 18 18 18 16 16 18
[289] 18 18 16 18 16 16 16 18 18 18 18 18 16 16 16 18 16 16 18 16 18 18 18 18 16 16 18 18 16 18 18 16
[321] 16 16 18 18 18 16 18 18 18 16 18 18 16 18 16 18 18 18 18 16 18 18 18 18 16 16 16 18 18 18 16 18
[353] 18 18 16 18 18 16 16 18 18 18 18 18 16 16 16 18 18 16 16 18 18 18 16 18 18 18 16 18 18 18 16 18
[385] 16 18 16 18 16 18 18 16 18 16 18 18 18 16 18 16 18 18 16 18 18 18 16 18 18 16 18 18 16 16 18 16
[417] 18 16 16 18 18 16 18 18 18 16 16 18 16 18 18 16 16 18 18 18 16 16 16 16 16 16 16 16 18 16 16 18
[449] 16 18 18 18 18 16 18 16 18 18 18 18 18 18 16 18 16 18 18 18 18 18 16 16 16 18 16 16 18 18 16 18
[481] 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 16 18 16 16 18 16 16 18 18 18 16 16 18 18 18 18 18
[513] 18 18 18 18 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 18 18 18 18 18 18 18 18 16 18 18 18 18
[545] 18 18 18 16 18 18 18 16 18 18 18 18 18 18 16 16 16 18 16 16 18 16 18 18 18 18 16 18 18 18 18 18
[577] 16 18 18 18 18 18 18 16 18 18 18 16 18 18 18 16 18 18 18 18 18 16 16 16 18 18 18 18 18 18
> plot(
+     ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Expresión génica (genes = 4897)

```
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_Transposed <- t(ExpGenTCGA_KIRC_Norm_Trans_Filt75)
> ExpGenTCGA_KIRC_Norm_Trans_Filt75_Transposed_PCA <- prcomp(ExpGenTCGA_KIRC_Norm_Trans_Filt75_Transpose
> str( ExpGenTCGA_KIRC_Norm_Trans_Filt75_Transposed_PCA )
```

```
List of 5
 $ sdev    : num [1:606] 36.9 23.9 17.4 15.8 13.2 ...
 $ rotation: num [1:4897, 1:606] 0.0201 0.0136 0.013 0.0159 0.0117 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
  .. ..$ : chr [1:606] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 16.2 12.3 12.8 12.3 11.9 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "AAMP" "AARS" "ABCA1" ...
 $ scale   : logi FALSE
 $ x       : num [1:606, 1:606] -32.54 -7.53 -5.93 -1.55 17.1 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:606] "TCGA-B0-5694-01A-11R-1541-07" "TCGA-CJ-4637-01A-02R-1325-07" "TCGA-CZ-4860-01A-
  .. ..$ : chr [1:606] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_RawData$vital_status)
> c(18, 16)[type]
  [1] 16 16 16 16 18 18 18 18 18 18 18 16 16 18 18 18 18 18 16 18 18 18 16 16 18 18 18 18 18 18 16 18
 [33] 16 18 18 16 18 16 16 18 18 16 18 18 18 16 18 16 18 16 16 18 18 18 16 18 18 18 18 18 18 18 18 18
 [65] 18 16 18 18 16 18 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 18 18 16 16 18 18 18 18 18 18 18
 [97] 16 18 18 18 18 18 16 18 16 16 16 18 18 16 16 18 16 18 16 18 18 16 16 18 18 18 16 16 18 18 16 16
[129] 18 16 18 18 18 18 18 18 18 18 18 18 16 18 18 18 18 18 18 18 18 18 18 18 18 16 16 16 16 16 18 16 16
[161] 18 18 16 16 18 18 16 18 16 18 16 16 18 16 18 16 16 18 18 18 18 16 16 16 18 18 18 18 18 18 18 18
[193] 16 18 18 18 18 18 18 16 16 18 18 18 16 16 18 16 18 18 16 18 18 18 18 18 18 18 16 16 18 16 18 16
[225] 18 18 18 18 16 18 18 18 18 16 16 18 18 18 18 18 18 16 18 16 18 18 18 16 18 16 18 18 16 18 18 18
[257] 16 18 16 18 18 16 18 16 16 18 16 16 18 18 18 18 16 18 18 16 16 18 18 18 18 18 18 18 16 16 18 18
[289] 18 18 16 18 16 16 16 18 18 18 18 18 16 16 16 18 16 16 18 16 18 18 18 18 16 16 18 18 16 18 18 16
[321] 16 16 18 18 18 16 18 18 18 16 18 18 16 18 16 18 18 18 18 16 18 18 18 18 16 16 16 18 18 18 16 18
[353] 18 18 16 18 18 16 16 18 18 18 18 18 16 16 16 18 18 16 16 18 18 18 16 18 18 18 16 18 18 18 16 18
[385] 16 18 16 18 16 18 18 16 18 16 18 18 18 16 18 16 18 18 16 18 18 18 16 18 18 16 18 18 16 16 18 16
[417] 18 16 16 18 18 16 18 18 18 16 16 18 16 18 18 16 16 18 18 18 16 16 16 16 16 16 16 18 16 16 18
[449] 16 18 18 18 18 16 18 16 18 18 18 18 18 18 16 18 16 18 18 18 18 16 16 16 18 16 16 18 18 16 18
[481] 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 16 18 16 16 18 16 16 18 18 18 16 16 18 18 18 18 18
[513] 18 18 18 18 18 18 16 18 18 18 16 18 18 18 18 16 18 18 18 18 18 18 18 18 18 18 16 18 18 18 18 18
[545] 18 18 18 16 18 18 18 16 18 18 18 18 18 18 16 16 16 18 16 16 18 16 18 18 18 18 16 18 18 18 18 18
[577] 16 18 18 18 18 18 18 16 18 18 18 16 18 18 18 16 18 18 18 18 18 16 16 16 18 18 18 18 18 18
> plot(
+     ExpGenTCGA_KIRC_Norm_Trans_Filt75_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Expresión proteica

Para obtener el máximo número de muestras con etiquetas solo podemos obtenerlas a partir del objeto que tiene las mismas muestras que en expresión génica (ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen), dado que los datos de Expresión proteica no vienen etiquetados como los otros.

```
> dim(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen)
[1] 177 474
```

Renombrar muestras

Tenemos que renombrar las muestras:

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Colnames <- colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSamp

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_ColnamesShort <- c()
for (j in ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Colnames){
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_ColnamesShort <- c(ExpProtTCGA_KIRC_RawData_woNA_SameSampE
}

ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed <- ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen
colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed) <- ExpProtTCGA_KIRC_RawData_woNA_SameSa
```

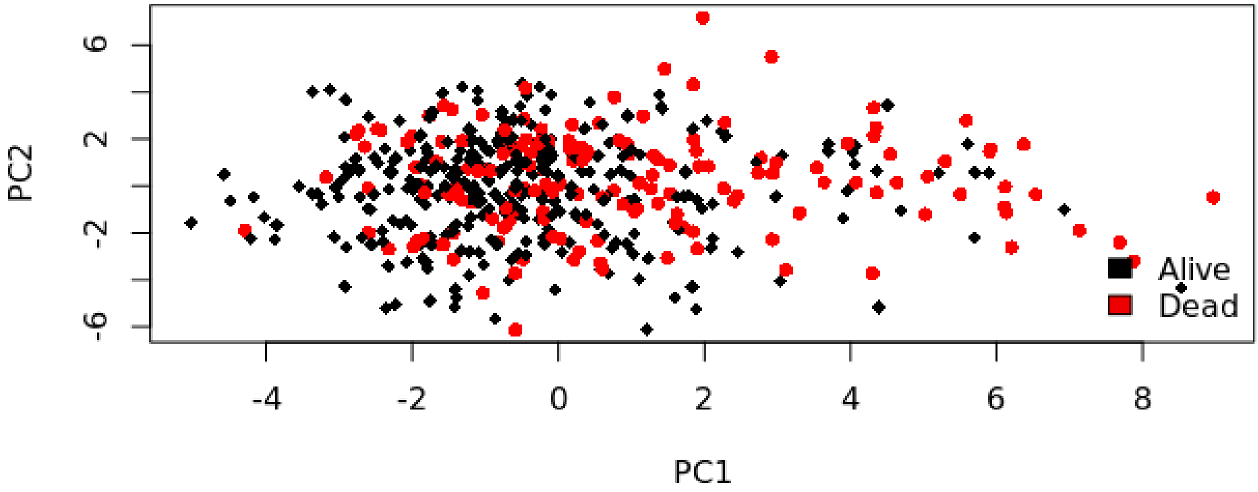
Obtener etiquetas

```
> x <- c()
> for (i in colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed)){
x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
}
```

```
> x
  [1] 161 546 202 114 235 595  25  52  38 593 539 429 504 476 232 254 187 183 151 582  37 490 418  69
 [25] 211 153 449 601 465 238 282  97  79 600 299 149 510 242 573 139 440  45 136 531 420 255 220 264
 [49] 154 375  17 113 495 119 422 356 360 540 269   3 162  15 109 484 311   6  63 165 441 591 456 558
 [73] 471 478 509 366 446 602  91 603 201 146 200 280 285 227 230 557  65 566  53 460  30 304 314 128
 [97] 486 330 454 487 389 315 362 122 323 173 433  14  55 286 237 428 129 547 251 125 379 367 586 512
[121]  22 390 290  24 178 432 105  42  64 244 385 413 213 333   7 239 369 338 199 553 427 174  16 206
[145] 249 450 163 361  90 209 303 472 378  67 497  80  60 402 599 451 434  54 208 339  96 358 184 277
[169] 159  19 488  71 605 166 555 372 581 156 545 261 514 523 233 312 344  78 388   8 167 305 506 340
[193] 363 343 324  27 185  21 147 240 574 415 597 519  23 386 551 143  29 475 309 470 528 435 123 228
[217] 234 587 565 307 224 403 308 431 458 384 257  81 405 336 348 508  28 382 364  59 461 189 351 563
[241] 580 477 444 560 443 533 326 325 252 583 168 567  98  70 371 483 164  73 507 517 421 522 260 491
[265] 396 535 266 293 349 559 468 321 300 245 354 525 104 130 198 133 205  51  36  48 157 377  72 500
[289] 135 289 150 482 345 395 392 320 467 197 448 193 101 549 265 302 110   1 281 414 256 529  84  92
[313] 452  83 322 207 222 527 329 267 117 505 248 502 594 332 492  61 116  40  93 438 409 543  86 295
[337] 127 176 283 480 121 572  13 331 137 219 425 106  77 250 365 408 180  26 411 196 328 577 459 171
[361] 182 370 564 218  76 453 160 158 347 469 296 335  57 319 571 270 346  10  44   4 195 532 437 496
[385] 550  82 503 102 100 120  56 383 542 596 279 537 466 337 493  75 112 186 568  47 439 455 350 190
[409] 276 464  66 387 534 103 548  99  50 380 353 397 231 313 499 511  74 148 259 423 352 598 179 236
[433] 424 229 445 273 217 124 520 417 394 463 436  85 327  68 357 407 381  20  35  32 538   5 556 442
[457] 268 401 398 294 175 515 274 579 263 155 406 368 216 132 118 272  31 275
> length(x)
[1] 474
> ExpGenTCGA_KIRC_RawData$sample[161]
[1] "TCGA-A3-3316-01A"
> colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed)[1]
[1] "TCGA-A3-3316-01A"
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed_Labels <- ExpGenTCGA_KIRC_RawData$vital_status[x]
```

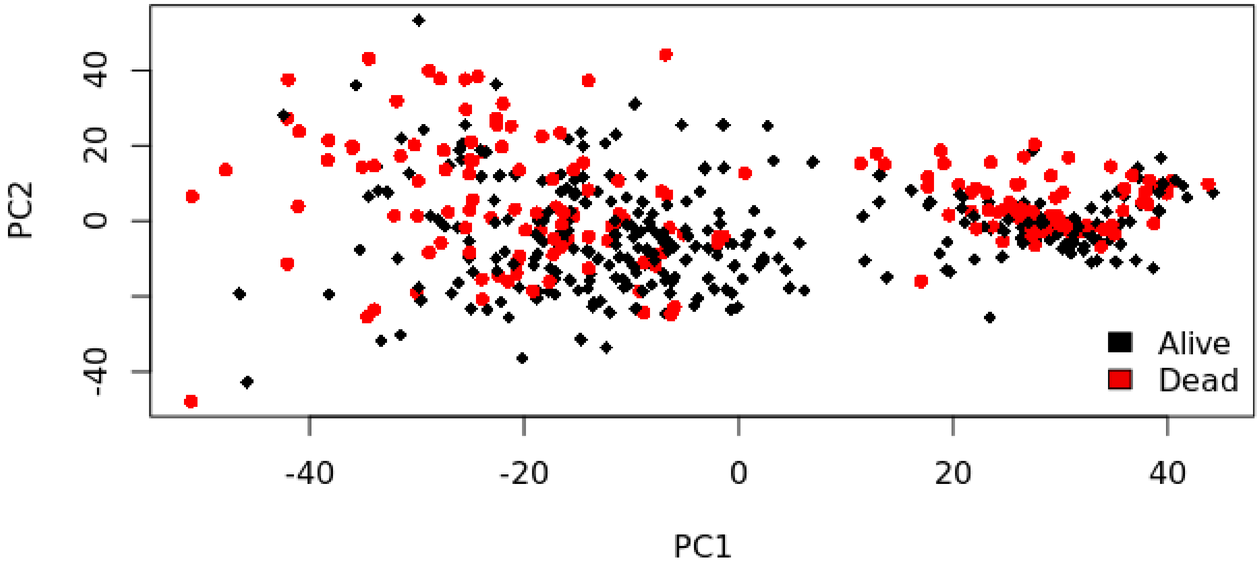
Análisis de primeras componentes

```
> ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Transposed <- t(ExpProtTCGA_KIRC_RawData_woNA_SameSampExp
> ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Transposed_PCA <- prcomp(ExpProtTCGA_KIRC_RawData_woNA_Sa
> str(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Transposed_PCA)
List of 5
 $ sdev    : num [1:177] 2.31 2.15 1.76 1.56 1.3 ...
 $ rotation: num [1:177, 1:177] 0.0529 0.0334 0.0445 -0.0271 -0.0291 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:177] "14-3-3_beta-R-V" "14-3-3_epsilon-M-C" "14-3-3_zeta-R-V" "4E-BP1_pS65-R-V" ...
  .. ..$ : chr [1:177] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:177] 0.08414 0.02667 0.03975 0.00587 0.06245 ...
  ..- attr(*, "names")= chr [1:177] "14-3-3_beta-R-V" "14-3-3_epsilon-M-C" "14-3-3_zeta-R-V" "4E-BP1_pS
 $ scale   : logi FALSE
 $ x       : num [1:474, 1:177] 0.322 -1.426 -1.8 -1.09 7.129 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:474] "TCGA-A3-3316-01A" "TCGA-BP-4970-01A" "TCGA-CJ-4884-01A" "TCGA-B8-A54D-01A" ...
  .. ..$ : chr [1:177] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen__Renamed_Labels)
> c(18, 16)[type]
  [1] 18 18 18 18 16 18 18 18 16 18 16 16 18 16 18 18 18 16 18 18 18 18 16 16 16 16 16 18 16 18 18 16
 [33] 18 16 18 18 18 16 18 18 16 18 18 18 18 18 16 16 16 16 18 16 18 16 16 18 18 18 18 16 18 18 18 18
 [65] 18 18 18 18 16 18 16 18 16 18 18 16 16 18 18 18 16 18 16 18 16 18 18 18 18 16 18 18 18 18 16 16
 [97] 18 16 16 16 16 18 18 18 18 18 16 18 16 16 18 18 18 18 18 18 16 16 18 18 18 18 18 16 18 16 16 16
[129] 18 16 16 16 18 16 18 18 18 18 18 18 16 16 18 16 18 18 16 18 18 18 16 16 18 18 18 18 18 18 16 18
[161] 18 18 16 18 18 16 16 16 16 16 18 18 18 18 18 18 18 16 18 18 18 16 18 18 18 18 18 18 16 16 16 16
[193] 18 18 18 18 16 18 18 18 18 18 18 16 16 18 18 18 18 16 18 18 16 18 16 18 16 18 18 18 16 16 16 18
[225] 18 18 16 18 18 18 18 18 18 18 18 18 18 18 16 16 18 18 16 16 16 18 16 18 18 18 18 18 18 18 16 16
[257] 16 16 16 18 18 18 18 18 18 18 18 16 18 16 18 16 18 18 18 18 18 16 18 18 16 16 16 16 16 18 18 18
[289] 18 18 18 18 16 18 16 16 18 18 18 16 18 18 16 16 16 16 18 16 18 18 18 18 18 18 16 18 16 18 18 16
[321] 18 18 16 16 18 18 16 18 18 18 18 16 18 18 18 16 16 16 18 18 18 18 16 18 18 16 18 16 16 16 16 18
[353] 18 18 18 18 18 16 18 16 18 16 16 18 18 18 16 18 16 18 18 16 18 18 16 18 16 18 18 16 18 18 16 16
[385] 18 16 18 18 18 18 18 16 18 18 18 18 18 18 18 18 18 18 18 18 16 18 18 18 16 18 16 16 18 16 16 18
[417] 16 18 18 18 18 16 16 18 18 18 16 18 18 16 18 18 18 16 18 16 18 16 18 18 16 16 18 18 18 18 18 16
[449] 18 18 18 18 18 18 18 16 16 18 16 16 18 18 18 18 18 16 18 18 18 18 16 18 16 18
> plot(
+     ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Metilación

```
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Transposed <- t(assay(MetTCGA_KIRC_RawData_woDupSamp
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Transposed_PCA <- prcomp(MetTCGA_KIRC_RawData_woDupS
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Transposed_PCA)
List of 5
 $ sdev    : num [1:483] 24.1 14.1 12.7 10.2 9.3 ...
 $ rotation: num [1:373382, 1:483] -2.59e-03 -2.16e-03 6.21e-05 6.35e-04 -1.12e-03 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:373382] "cg00000029" "cg00000165" "cg00000236" "cg00000289" ...
  .. ..$ : chr [1:483] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:373382] 0.513 0.19 0.897 0.703 0.535 ...
  ..- attr(*, "names")= chr [1:373382] "cg00000029" "cg00000165" "cg00000236" "cg00000289" ...
 $ scale   : logi FALSE
 $ x       : num [1:483, 1:483] -0.611 -22.183 -22.077 -6.058 38.29 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:483] "TCGA-BP-4993-01A-02D-1418-05" "TCGA-CZ-5982-01A-11D-1670-05" "TCGA-B0-4703-01A-
  .. ..$ : chr [1:483] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA$vital_status)
> c(18, 16)[type]
  [1] 18 18 16 18 18 16 18 18 18 16 18 18 18 16 18 16 18 18 18 18 18 18 18 18 18 16 18 18 18 16 16 16
 [33] 18 18 18 18 18 16 18 16 16 16 18 18 18 18 18 18 18 18 18 18 16 16 18 18 16 18 18 18 18 18 18 18
 [65] 18 18 18 18 16 16 18 18 18 18 16 18 16 16 18 16 16 18 18 18 18 16 16 16 16 18 18 18 18 18 16 16
 [97] 18 16 18 16 16 18 16 18 18 16 18 18 16 16 18 16 18 16 18 18 16 16 16 18 18 16 16 18 16 18 18 18
[129] 18 18 18 18 18 18 18 16 16 18 16 16 18 18 16 18 16 18 16 18 18 18 18 18 16 16 18 18 16 18 16 16
[161] 18 18 18 16 18 16 16 16 18 18 18 18 18 18 18 18 18 18 16 18 16 18 18 16 18 16 16 18 18 18 18 16
[193] 18 16 18 18 18 18 18 18 18 18 16 18 18 18 18 18 18 18 18 16 16 18 16 16 18 18 16 16 16 16 16 16
[225] 18 18 16 18 18 16 18 18 18 16 16 18 18 18 16 18 18 16 18 18 18 18 18 18 18 18 16 16 18 16 16 18
[257] 16 18 16 18 16 16 16 18 18 18 18 18 16 16 16 16 18 16 18 16 16 16 18 18 16 18 18 18 18 18 18 16
[289] 16 18 18 18 18 16 18 16 18 18 16 18 18 16 16 18 16 18 18 18 16 18 16 16 18 18 18 16 16 18 18 18
[321] 16 18 18 18 16 16 16 16 18 18 18 16 16 16 18 18 16 18 16 16 16 18 18 18 16 18 18 16 18 18 18 16
[353] 16 16 18 18 16 16 16 16 18 16 18 18 16 18 16 18 18 18 16 18 18 16 18 18 16 18 16 18 16 18 16 18
[385] 16 18 18 18 16 16 16 18 18 18 16 16 18 16 18 18 18 16 18 18 16 18 16 18 16 16 16 18 18 16 18 16
[417] 18 18 18 16 16 16 16 18 18 18 18 16 16 18 16 16 18 18 18 18 18 18 18 18 18 16 16 18 18 16 18 16
[449] 16 16 16 18 18 18 18 18 18 16 18 18 18 16 18 18 16 18 18 16 18 16 16 16 18 18 18 16 18 18 18 18
[481] 16 18 18
> plot(
+     MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Transposed_PCA$x,
+     col = type,
+     pch = c(18, 16)[type],
+     cex = 1.0)
> legend(
+     "bottomright",
+     bty = "n",
+     c("Alive", "Dead"),
+     fill = c("black", "red2"),
+     cex = 1.0)
```



Exportar todos los objetos que se han creado y procesado

Modelo de Expresión génica (genes = 238)

Base de datos

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed, file = "E
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, file
```

3. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Train, fil
```

Etiquetas

Tipo Caracteres

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test, file = "
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Trai

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train, file =
```

Tipo Factor

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNum
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Trai

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNu
```

Modelo de Expresión génica (genes = 4897)

Base de datos

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed, file = "ExpGenTC
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, file = "Exp
```

3. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train, file = "Ex
```

Etiquetas

Tipo Caracteres

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test, file = "ExpGenT
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train, file = "ExpGen
```

Tipo Factor

1. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNu

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb, file
```

2. ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorN

```
save(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb, fil
```

Modelo de Expresión proteica

Base de datos

1. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed, file = "ExpProtTCGA_K
```

2. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Test

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Test, file = "ExpProtT
```

3. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Train

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Train, file = "ExpProt
```

Etiquetas

Tipo Caracteres

1. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test, file = "ExpProtTCGA_
```

2. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train, file = "ExpProtTCGA_
```

Tipo Factor

1. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb, file = "E
```

2. ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNum

```
save(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb, file = "
```

Modelo de metilación

Base de datos

1. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed, file = "MetT
```

2. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Test, file =
```

3. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Train, file =
```

Etiquetas

Tipo Caracteres

1. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test, file = "Met
```

2. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train, file = "Me
```

Tipo Factor

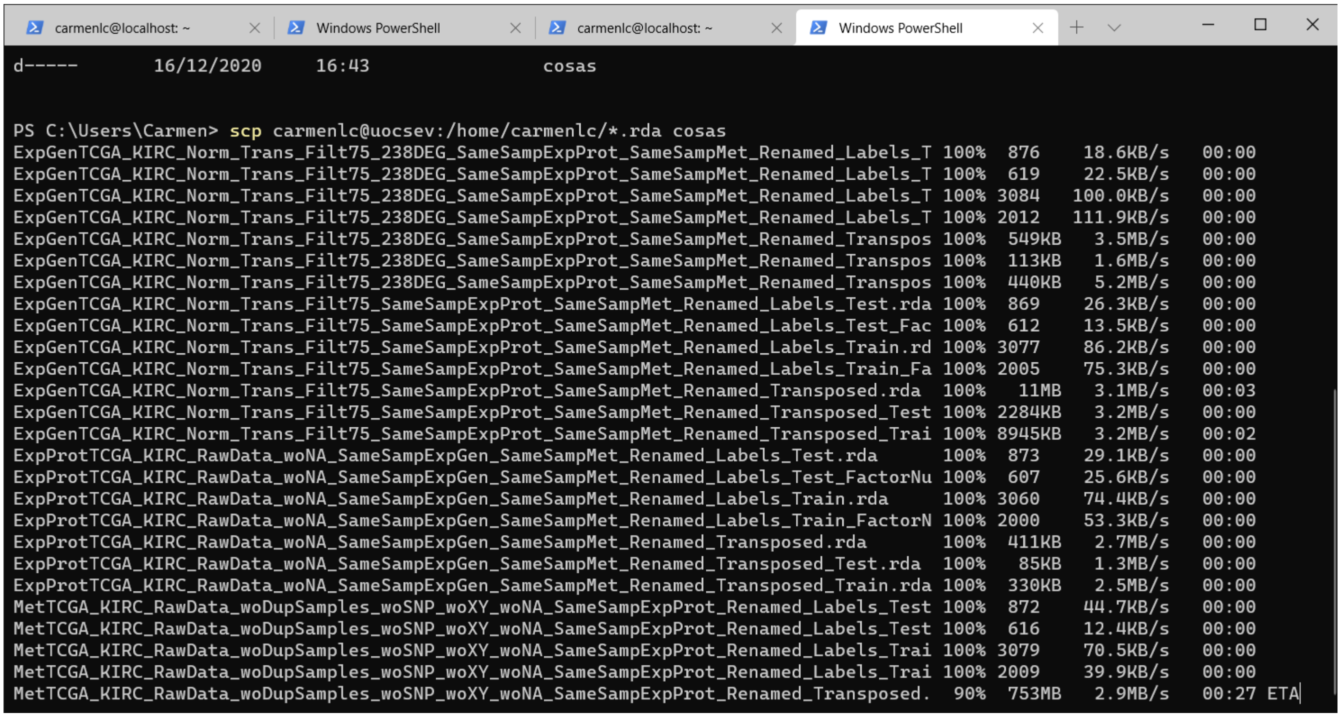
1. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test_

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test_FactorNumb,
```

2. MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train_FactorNumb,
```

Mover objetos del servidor al local



```
d-----        16/12/2020     16:43                cosas


PS C:\Users\Carmen> scp carmenlc@uocsev:/home/carmenlc/*.rda cosas
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_T 100%  876     18.6KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_T 100%  619     22.5KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_T 100% 3084    100.0KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_T 100% 2012    111.9KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transpos 100%  549KB    3.5MB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transpos 100%  113KB    1.6MB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transpos 100%  440KB    5.2MB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test.rda 100%  869     26.3KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_Fac 100%  612     13.5KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train.rd 100% 3077     86.2KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_Fa 100% 2005     75.3KB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed.rda  100%   11MB    3.1MB/s   00:03
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test 100% 2284KB    3.2MB/s   00:00
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Trai 100% 8945KB    3.2MB/s   00:02
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test.rda      100%  873     29.1KB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNu 100%  607     25.6KB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train.rda     100% 3060     74.4KB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorN 100% 2000     53.3KB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed.rda       100%  411KB    2.7MB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Test.rda  100%   85KB    1.3MB/s   00:00
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Train.rda 100%  330KB    2.5MB/s   00:00
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test 100%  872     44.7KB/s   00:00
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test 100%  616     12.4KB/s   00:00
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Trai 100% 3079     70.5KB/s   00:00
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Trai 100% 2009     39.9KB/s   00:00
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed.  90%  753MB    2.9MB/s   00:27 ETA
```

Creación de modelos

Ver: https://tensorflow.rstudio.com/tools/tfruns/overview/

The tfruns package provides a suite of tools for tracking, visualizing, and managing TensorFlow training runs and experiments from R:

- Track the hyperparameters, metrics, output, and source code of every training run.
- Compare hyperparmaeters and metrics across runs to find the best performing model.
- Automatically generate reports to visualize individual training runs or comparisons between runs.
- No changes to source code required (run data is automatically captured for all Keras and TF Estimator models).

Instalación del paquete

You can install the tfruns package from CRAN as follows:

```
install.packages("tfruns")
```

The package is intended to be used with the keras and/or the tfestimators packages, both of which provide higher level interfaces to TensorFlow from R. These packages can be installed with:

```
install.packages("keras")
install.packages("tfestimators")
```

Para utilizar este paquete necesitamos crear un script de R en el que pondremos nuestro código de entrenamiento. Lo bueno de este paquete es que tras realizar un entrenamiento, aparecerá un informe de este en pantalla si se está utilizando un entorno interactivo de R y las metricas se capturarán automáticamente.

Modelo de Expresión génica (genes = 238)

Creando el script de R

Hemos guardado todos los objetos para no tener que realizar todo el procesado de datos y simplemente cargarlos.

Script que se guardará como `ExpGen238modelscript.R`:

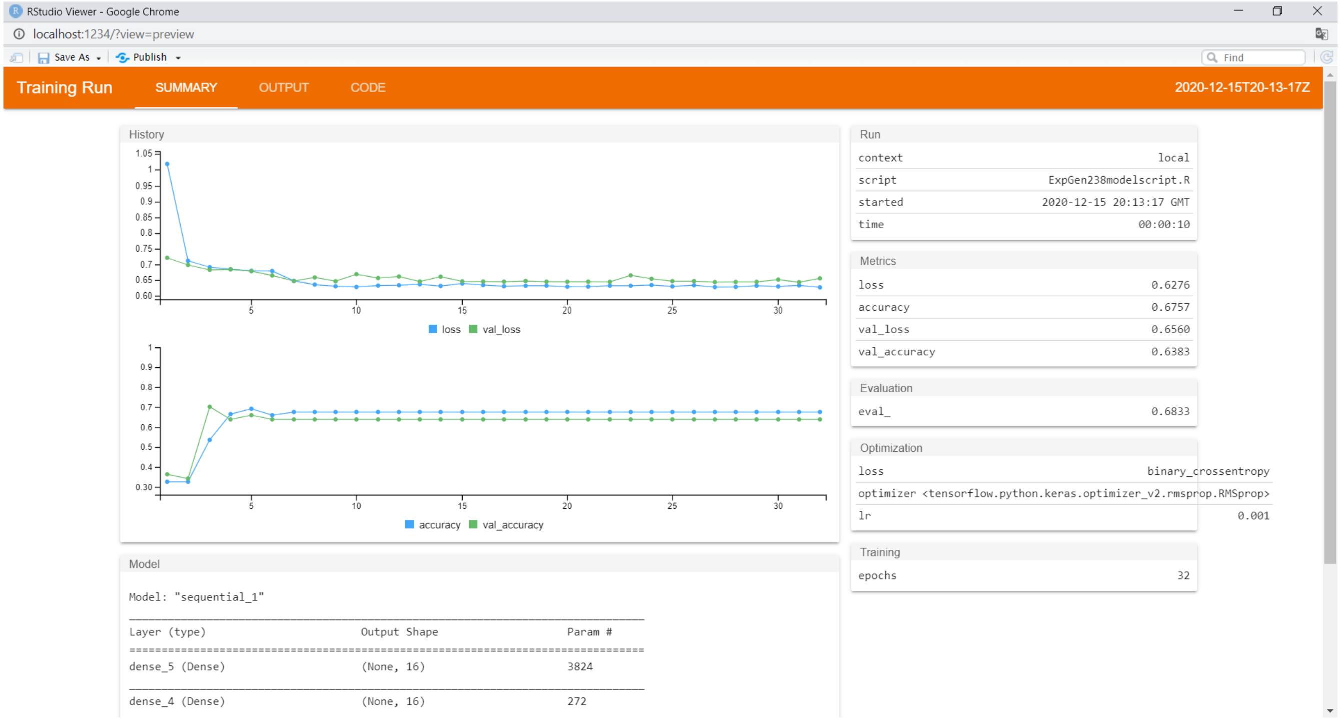
```
# Cargando los archivos

# test_x
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Test.rda"

# train_x
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Train.rda

# test_y
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNu

# train_y
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorN

library(lattice)
library(ggplot2)
library(keras)
library(caret)

# Definición del modelo ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG
Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed <- keras_model_sequer
  layer_dense(units = 16, activation = "relu", input_shape = c(238)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed %>% fit(E

plot(history)

score <- Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed %>% evaluat
  ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, ExpGenT
  verbose = 0
)

cat('Test loss:', score[[1]], '\n')
cat('Test accuracy:', score[[2]], '\n')
```

In the following sections we'll describe the various capabilities of tfruns. Our training script (ExpGen238modelscript.R) trains a Keras model to classify vital_status of patients.

To train a model with tfruns, just use the `training_run()` function in place of the `source()` function to execute your R script. For example:

```
library(tfruns)
training_run("ExpGen238modelscript.R")
```

When training is completed, a summary of the run will automatically be displayed if you are within an interactive R session:



Run

context	local
script	ExpGen238modelscript.R
started	2020-12-15 20:13:17 GMT
time	00:00:10

Metrics

loss	0.6276
accuracy	0.6757
val_loss	0.6560
val_accuracy	0.6383

Evaluation

eval_	0.6833

Optimization

loss	binary_crossentropy
optimizer	<tensorflow.python.keras.optimizer_v2.rmsprop.RMSprop>
lr	0.001

Training

epochs	32

Model

```
Model: "sequential_1"

________________________________________________________________________________
Layer (type)                        Output Shape                    Param #
================================================================================
dense_5 (Dense)                     (None, 16)                      3824
________________________________________________________________________________
dense_4 (Dense)                     (None, 16)                      272
```

The metrics and output of each run are automatically captured within a run directory which is unique for each run that you initiate. Note that for Keras and TF Estimator models this data is captured automatically (no changes to your source code are required).

You can call the latest_run() function to view the results of the last run (including the path to the run directory which stores all of the run's output):

```
latest_run()
$ run_dir               : chr "runs/2020-12-15T20-13-17Z"
 $ eval_                : num 0.683
 $ metric_loss          : num 0.628
 $ metric_accuracy      : num 0.676
 $ metric_val_loss      : num 0.656
 $ metric_val_accuracy: num 0.638
 $ epochs               : int 32
 $ epochs_completed     : int 32
 $ metrics              : chr "(metrics data frame)"
 $ model                : chr "(model summary)"
 $ loss_function        : chr "binary_crossentropy"
 $ optimizer            : chr "<tensorflow.python.keras.optimizer_v2.rmsprop.RMSprop>"
 $ learning_rate        : num 0.001
 $ script               : chr "ExpGen238modelscript.R"
 $ start                : POSIXct[1:1], format: "2020-12-15 20:13:17"
 $ end                  : POSIXct[1:1], format: "2020-12-15 20:13:28"
 $ completed            : logi TRUE
 $ output               : chr "(script ouptut)"
 $ source_code          : chr "(source archive)"
 $ context              : chr "local"
 $ type                 : chr "training"
 $ NA.                  : num 0.621
```

The run directory used in the example above is "runs/2020-12-15T20-13-17Z". Run directories are by default generated within the “runs” subdirectory of the current working directory, and use a timestamp as the name of the run directory. You can view the report for any given run using the view_run() function:

```
view_run("runs/2020-12-15T20-13-17Z")
```

Resultados de los modelos

Es muy extraño que cambiando arquitecturas, incluso poniendo solo dos capas y añadiendo dropout el resultado de la precisión del modelo siga siendo el mismo (sobre el 67%). Vamos a ver cuáles son las predicciones del modelo por si hay algún problema:

```
> predict(Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed,ExpGenTCGA_
           [,1]
 [1,] 0.9149444
 [2,] 0.9183401
 [3,] 0.9166617
 [4,] 0.9275434
 [5,] 0.9490658
 [6,] 0.9190948
 [7,] 0.9068053
 [8,] 0.9331865
 [9,] 0.9094759
[10,] 0.8935333
[11,] 0.9020542
[12,] 0.9259428
[13,] 0.9137262
[14,] 0.9206098
[15,] 0.9329575
[16,] 0.9282410
[17,] 0.9244517
[18,] 0.9257691
[19,] 0.9212096
[20,] 0.9243253
[21,] 0.9101219
[22,] 0.9339755
[23,] 0.9186105
[24,] 0.9155498
[25,] 0.9194268
[26,] 0.9151125
[27,] 0.9336702
[28,] 0.9302230
[29,] 0.9422885
[30,] 0.9181346
[31,] 0.9315906
[32,] 0.9310535
[33,] 0.9310513
[34,] 0.8539923
[35,] 0.9314967
[36,] 0.9009373
[37,] 0.9096189
[38,] 0.9085835
[39,] 0.9276797
[40,] 0.9115001
```

```
[41,] 0.9289750
[42,] 0.9394258
[43,] 0.9405989
[44,] 0.9227557
[45,] 0.9296468
[46,] 0.9282748
[47,] 0.8985977
[48,] 0.9322637
[49,] 0.9231887
[50,] 0.9160763
[51,] 0.9095685
[52,] 0.9150640
[53,] 0.9271596
[54,] 0.9291655
[55,] 0.8605986
[56,] 0.9225242
[57,] 0.9228839
[58,] 0.9367518
> str(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNu
 num [1:58] 1 1 1 1 1 1 0 0 1 1 ...
> table(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorN
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb
 0  1
22 36
> 36/58
[1] 0.6206897
```

Observamos que el modelo siempre tiende a elegir un resultado cercano a 1 (Vivo), en todas y cada una de las predicciones y es por ello que el porcentaje es de un 62% porque en nuestro dataset de prueba hay un 62% de pacientes vivos. En el dataset de entrenamiento hay un 67% de pacientes vivos.

Vamos a hacer una red más grande. En este caso nuestro objetivo no es aumentar el porcentaje de acierto (aunque también), sino ver si la red es capaz de predecir algún caso como 0 (muerto), esto nos enseñará que la red está intentando aprender algo.

Con un modelo:

Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed <- keras_model_sequential() %>% layer_dense(units = 200, activation = “relu”, input_shape = c(238)) %>% layer_dense(units = 150, activation = “relu”) %>% layer_dense(units = 100, activation = “relu”) %>% layer_dense(units = 50, activation = “relu”) %>% layer_dense(units = 25, activation = “relu”) %>% layer_dense(units = 1, activation = “sigmoid”)

```
> predict(Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed,ExpGenTCGA_
           [,1]
 [1,] 0.8109230
 [2,] 0.8244340
 [3,] 0.8271601
 [4,] 0.8434394
 [5,] 0.8154917
 [6,] 0.8270776
 [7,] 0.8318338
 [8,] 0.8401296
 [9,] 0.8291189
[10,] 0.8379909
```

```
[11,] 0.8143069
[12,] 0.8347751
[13,] 0.8371578
[14,] 0.8364050
[15,] 0.8462929
[16,] 0.8461750
[17,] 0.8058777
[18,] 0.8264309
[19,] 0.8383197
[20,] 0.8347566
[21,] 0.7986511
[22,] 0.8437719
[23,] 0.8298242
[24,] 0.8332751
[25,] 0.8163152
[26,] 0.8363693
[27,] 0.8370081
[28,] 0.8342853
[29,] 0.8434381
[30,] 0.8193663
[31,] 0.8475674
[32,] 0.8428571
[33,] 0.8313342
[34,] 0.7747893
[35,] 0.8430301
[36,] 0.8231998
[37,] 0.8354641
[38,] 0.8458142
[39,] 0.8128962
[40,] 0.8135376
[41,] 0.8285904
[42,] 0.8423585
[43,] 0.8457391
[44,] 0.8116895
[45,] 0.8509319
[46,] 0.8316950
[47,] 0.8202958
[48,] 0.8009056
[49,] 0.8322530
[50,] 0.8034012
[51,] 0.7949884
[52,] 0.8253350
[53,] 0.8464656
[54,] 0.8577033
[55,] 0.7587770
[56,] 0.8236715
[57,] 0.8356177
[58,] 0.8208525
```

También hemos hecho pruebas con una red más pequeña, por ejemplo:

Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed <- keras_model_sequential() %>% layer_dense(units = 10, activation = “relu”, input_shape = c(238)) %>% layer_dense(units = 4, activation = “relu”) %>% layer_dense(units = 1, activation = “sigmoid”)

En este caso lo que ocurre cuando observamos las predicts es que se repite casi siempre el mismo número, por lo que la red sigue sin aprender:

```
> predict(Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_SameSampMet_Renamed,ExpGenTCGA_
           [,1]
 [1,] 0.5037997
 [2,] 0.5037997
 [3,] 0.5037997
 [4,] 0.5037997
 [5,] 0.5037997
 [6,] 0.5037997
 [7,] 0.5037997
 [8,] 0.5037997
 [9,] 0.5037997
[10,] 0.5269402
[11,] 0.5037997
[12,] 0.5037997
[13,] 0.5037997
[14,] 0.5037997
[15,] 0.5037997
[16,] 0.5037997
[17,] 0.5037997
[18,] 0.5037997
[19,] 0.5037997
[20,] 0.5037997
[21,] 0.5037997
[22,] 0.5037997
[23,] 0.5037997
[24,] 0.5037997
[25,] 0.5037997
[26,] 0.5037997
[27,] 0.5037997
[28,] 0.5037997
[29,] 0.5037997
[30,] 0.5037997
[31,] 0.5037997
[32,] 0.5037997
[33,] 0.5037997
[34,] 0.5171638
[35,] 0.5037997
[36,] 0.5037997
[37,] 0.5037997
[38,] 0.5037997
[39,] 0.5037997
[40,] 0.5037997
[41,] 0.5037997
[42,] 0.5037997
[43,] 0.5037997
[44,] 0.5037997
[45,] 0.5037997
[46,] 0.5037997
[47,] 0.5037997
[48,] 0.5037997
[49,] 0.5037997
```

```
[50,] 0.5037997
[51,] 0.5037997
[52,] 0.5037997
[53,] 0.5037997
[54,] 0.5037997
[55,] 0.5107551
[56,] 0.5037997
[57,] 0.5037997
[58,] 0.5037997
```

Con un entrenamiento con 100 epochs, no parece que la red aprenda ya que seguimos obteniendo valores de las predicciones cercanos a 1. Además con tantas iteraciones la red debería ser capaz de alcanzar el overfitting, por lo que:

- O los grupos no están lo suficientemente diferenciados.
- La red no tiene suficientes datos de entrada (muestras) como para ser capaz de aprender.

Modelo de Expresión génica (genes = 4897)

Creando Script

Hemos guardado todos los objetos para no tener que realizar todo el procesado de datos y simplemente cargarlos.

Script que se guardará como `ExpGen4897modelscript.R`:

```
# Cargando los archivos

# test_x
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test.rda")

# train_x
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train.rda")

# test_y
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb.rda"

# train_y
load("ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb.rda

library(lattice)
library(ggplot2)
library(keras)
library(caret)

# Definición del modelo ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG
Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed <- keras_model_sequential()
  layer_dense(units = 16, activation = "relu", input_shape = c(4897)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed %>% fit(ExpGenTC

plot(history)

score <- Model_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed %>% evaluate(
  ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, ExpGenTCGA_KIR
  verbose = 0
)

cat('Test loss:', score[[1]], '\n')
cat('Test accuracy:', score[[2]], '\n')
```

Resultados

Aunque utilicemos distintas arquitecturas seguimos obteniendo ese porcentaje de precisión del 67%. La red sigue sin aprender aunque utilicemos los datos de expresión génica con todos los genes (solo con el filtrado) sin haber realizado el análisis de expresión diferencial.

Modelo de Expresión proteica

Creando script de R

El script utilizado es `ExpProt177modelscript.R`:

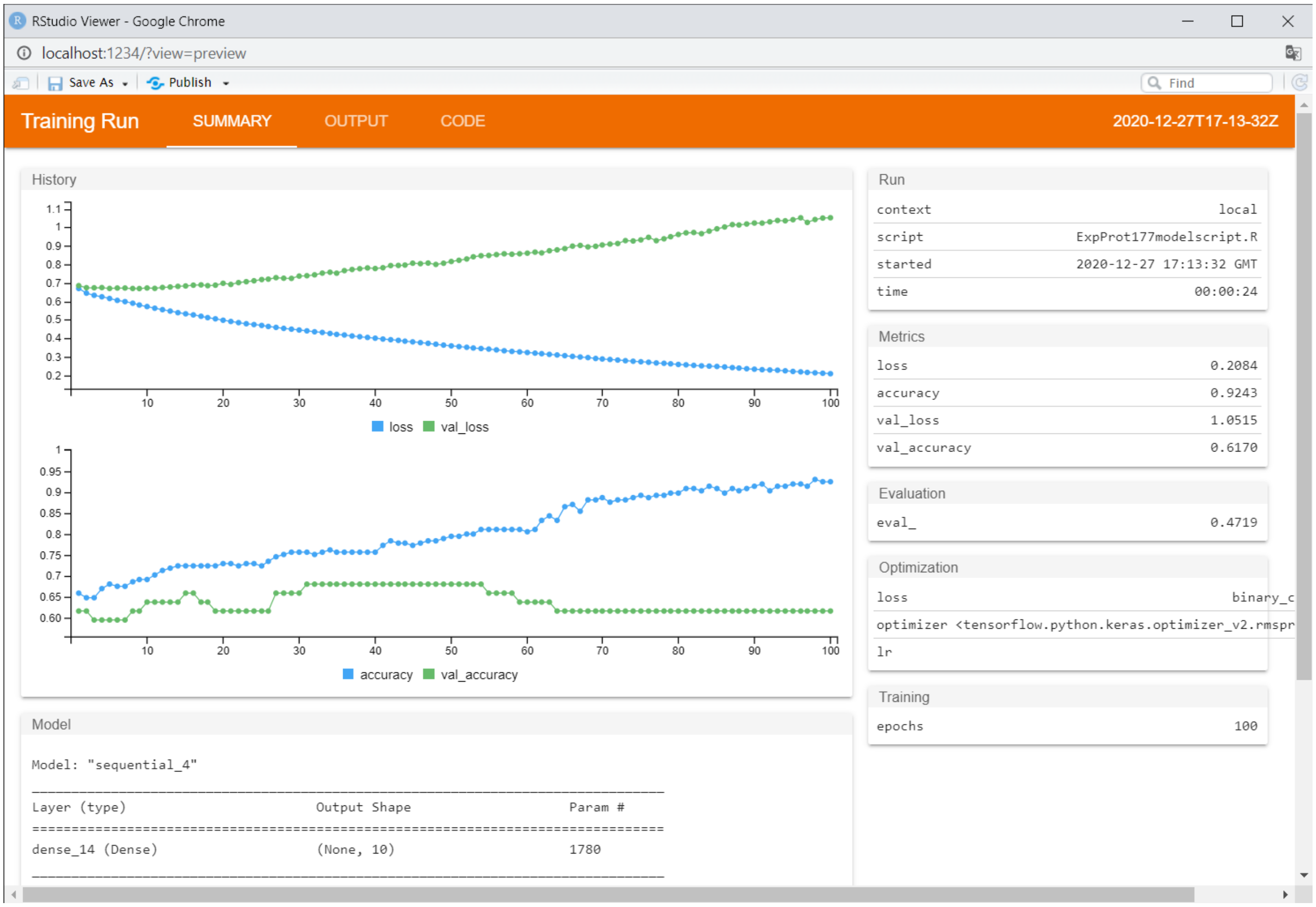
```
# Cargando los archivos

# test_x
load("ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Test.rda")

# train_x
load("ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Transposed_Train.rda")

# test_y
load("ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb.rda")

# train_y
load("ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb.rda")

library(lattice)
library(ggplot2)
library(keras)
library(caret)

# Definición del modelo ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed
Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed <- keras_model_sequential() %>%
  layer_dense(units = 10, activation = "relu", input_shape = c(177)) %>%
  layer_dense(units = 4, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

summary(Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)

Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed %>% fit(ExpProtTCGA_KI

plot(history)

score <- Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed %>% evaluate(ExpProtTCG

)

cat('Test loss:', score[[1]], '\n')
cat('Test accuracy:', score[[2]], '\n')
```

Resultados

En este caso, con una red:

Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed <- keras_model_sequential() %>% layer_dense(units = 10, activation = "relu", input_shape = c(177)) %>% layer_dense(units = 4, activation = "relu") %>% layer_dense(units = 1, activation = "sigmoid")

Si que observamos el overfitting que tanto buscábamos:



Run

```
context                                   local
script                   ExpProt177modelscript.R
started                 2020-12-27 17:13:32 GMT
time                                   00:00:24
```

Metrics

```
loss                                     0.2084
accuracy                                 0.9243
val_loss                                 1.0515
val_accuracy                             0.6170
```

Evaluation

```
eval_                                    0.4719
```

Optimization

```
loss                                   binary_c
optimizer <tensorflow.python.keras.optimizer_v2.rmspr
lr
```

Training

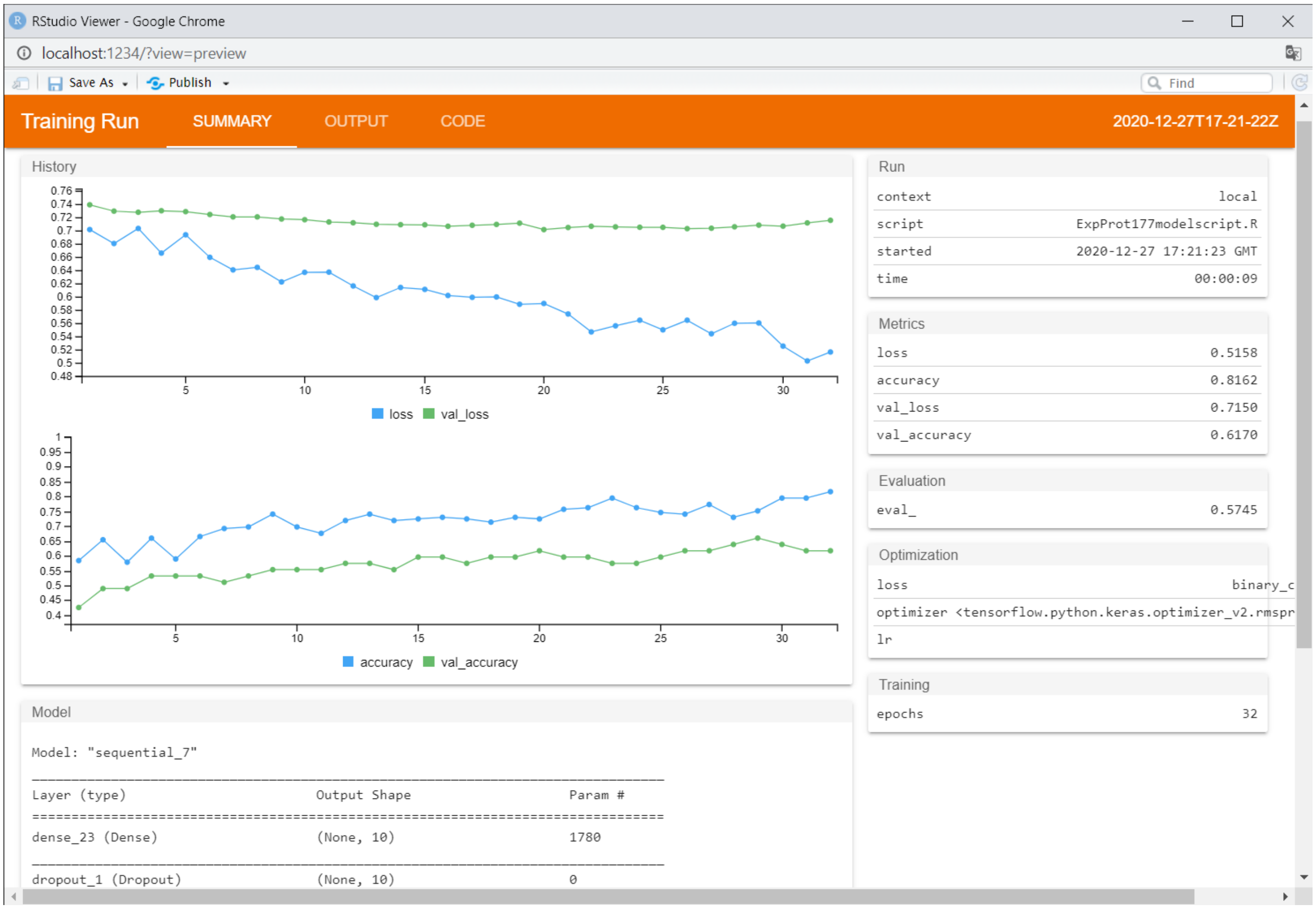
```
epochs                                      100
```

Model

```
Model: "sequential_4"
________________________________________________________________________________
Layer (type)                        Output Shape                        Param #
================================================================================
dense_14 (Dense)                    (None, 10)                          1780
________________________________________________________________________________
```

Vamos a añadirle dos capas de dropout:

Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed <- keras_model_sequential() %>% layer_dense(units = 10, activation = "relu", input_shape = c(177)) %>% layer_dropout(0.2) %>% layer_dense(units = 4, activation = "relu") %>% layer_dropout(0.2) %>% layer_dense(units = 1, activation = "sigmoid")



Run

context	local
script	ExpProt177modelscript.R
started	2020-12-27 17:21:23 GMT
time	00:00:09

Metrics

loss	0.5158
accuracy	0.8162
val_loss	0.7150
val_accuracy	0.6170

Evaluation

eval_	0.5745

Optimization

loss	binary_c
optimizer <tensorflow.python.keras.optimizer_v2.rmspr	
lr	

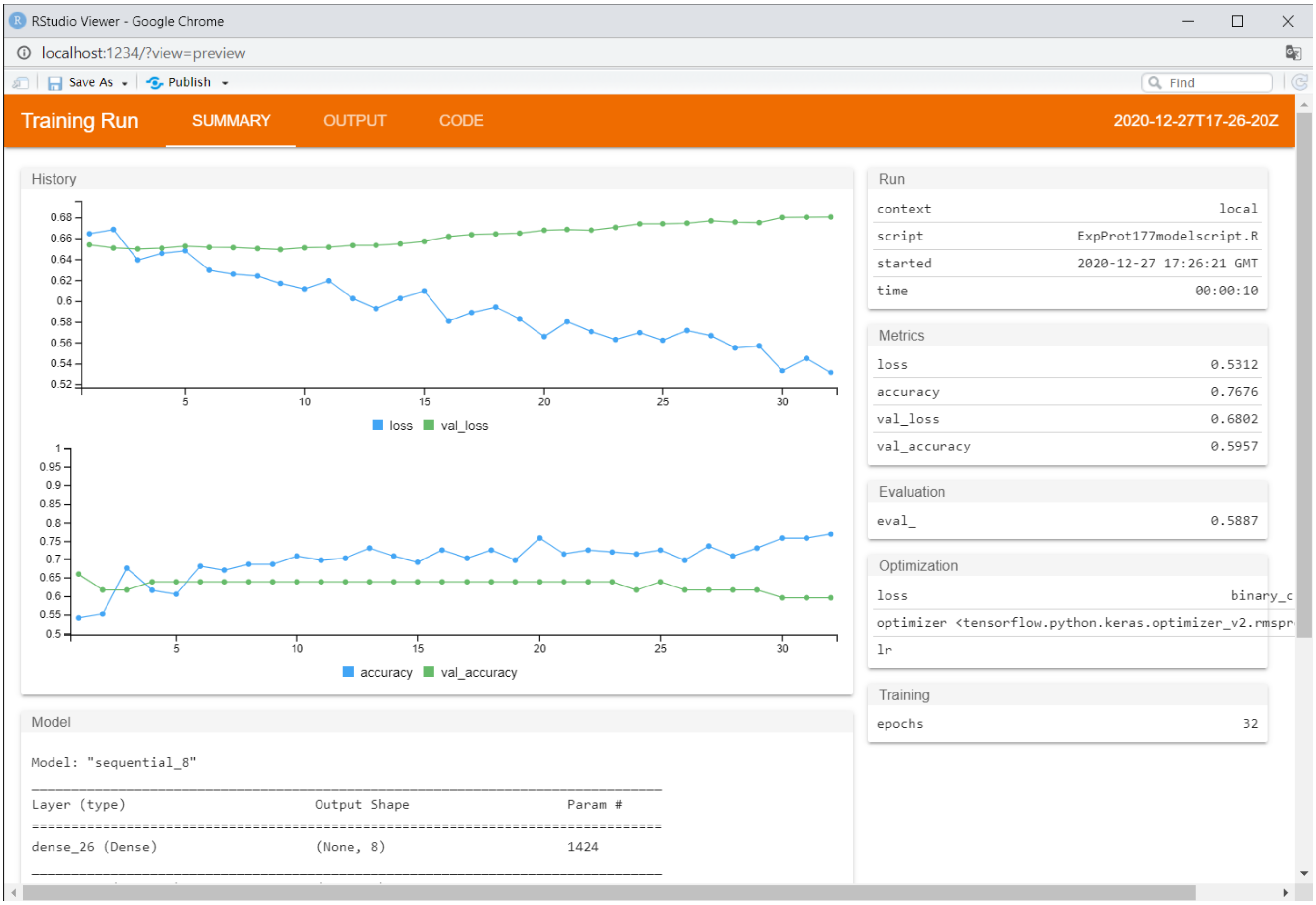
Training

epochs	32

Model

```
Model: "sequential_7"
________________________________________________________________________________
Layer (type)                        Output Shape                      Param #
================================================================================
dense_23 (Dense)                    (None, 10)                        1780
________________________________________________________________________________
dropout_1 (Dropout)                 (None, 10)                        0
```

Sigue habiendo overfitting, vamos a hacer la red un poco más pequeña. Las precisiones de la clasificación se van acercando pero los loss siguen siendo contrapuestos.



Run

context	local
script	ExpProt177modelscript.R
started	2020-12-27 17:26:21 GMT
time	00:00:10

Metrics

loss	0.5312
accuracy	0.7676
val_loss	0.6802
val_accuracy	0.5957

Evaluation

eval_	0.5887

Optimization

loss	binary_c
optimizer	<tensorflow.python.keras.optimizer_v2.rmspr
lr	

Training

epochs	32

Model

```
Model: "sequential_8"

________________________________________________________________________________
Layer (type)                        Output Shape                    Param #
================================================================================
dense_26 (Dense)                    (None, 8)                       1424
________________________________________________________________________________
```

Para:

Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed <- keras_model_sequential() %>% layer_dense(units = 8, activation = "relu", input_shape = c(177)) %>% layer_dropout(0.2) %>% layer_dense(units = 2, activation = "relu") %>% layer_dropout(0.2) %>% layer_dense(units = 1, activation = "sigmoid")

Modelo de metilación

Creando el script

El script utilizado es `Met373382modelscript.R`:

```
# Cargando los archivos

# test_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Test.rda")

# train_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Transposed_Train.rda")

# test_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Test_FactorNumb.
# train_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed_Labels_Train_FactorNumb

library(lattice)
library(ggplot2)
library(keras)
library(caret)

# Definición del modelo MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed
Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed <- keras_model_sequentia
  layer_dense(units = 8, activation = "relu", input_shape = c(373382)) %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 2, activation = "relu") %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 1, activation = "sigmoid")

summary(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed)

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed %>% fit(MetT

plot(history)

score <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed %>% evaluate(M
                                                                                                 verbose =
)

cat('Test loss:', score[[1]], '\n')
cat('Test accuracy:', score[[2]], '\n')
```

Resultados

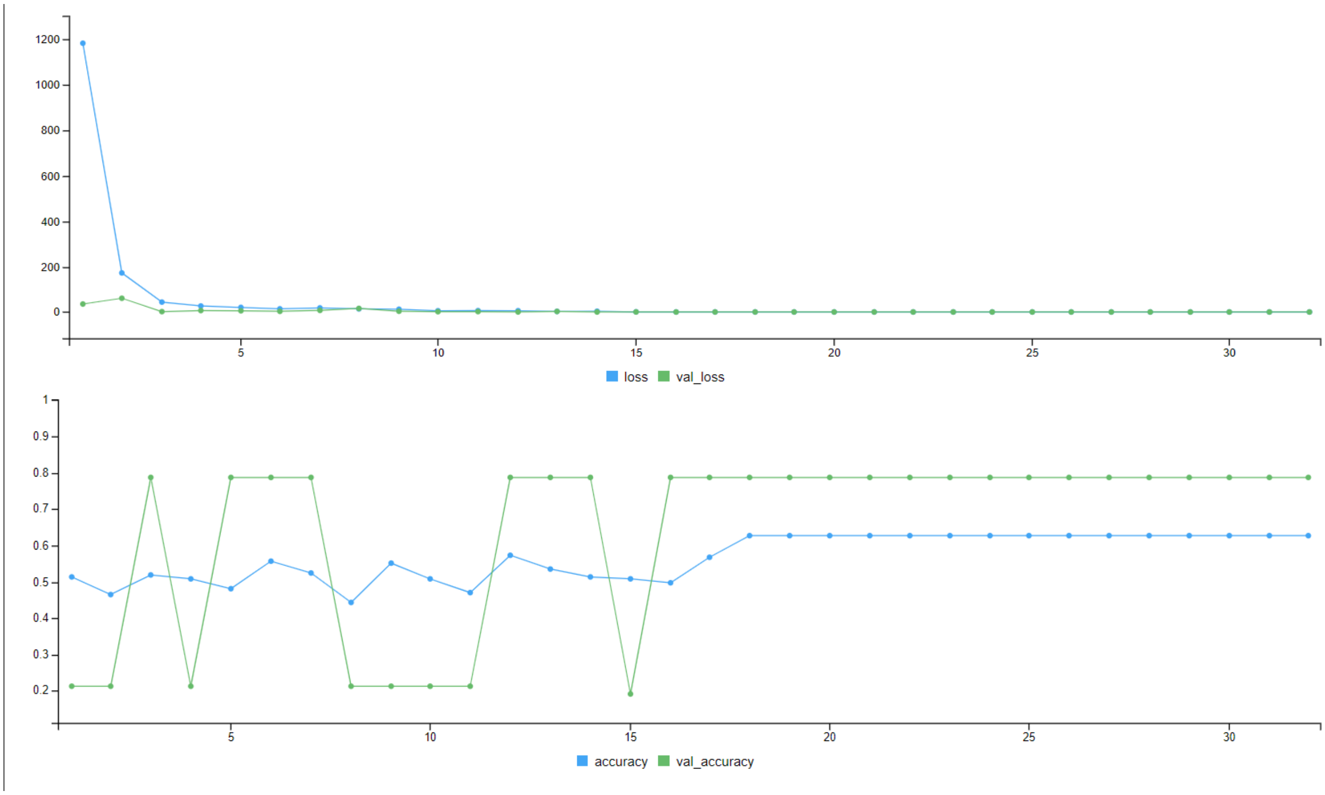
Intentando buscar el overfitting creamos el siguiente modelo:

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed <- keras_model_sequential() %>% layer_dense(units = 200, activation = "relu", input_shape = c(373382)) %>% layer_dropout(0.2) %>% layer_dense(units = 100, activation = "relu") %>% layer_dropout(0.2) %>% layer_dense(units = 50, activation = "relu") %>% layer_dropout(0.2) %>% layer_dense(units = 1, activation = "sigmoid")

Con el que obtenemos finalmente un score de:

```
> score
     loss  accuracy
0.6720954 0.6034483
> history
```

```
Final epoch (plot to see history):
        loss: 0.6667
    accuracy: 0.627
    val_loss: 0.6078
val_accuracy: 0.7872
```



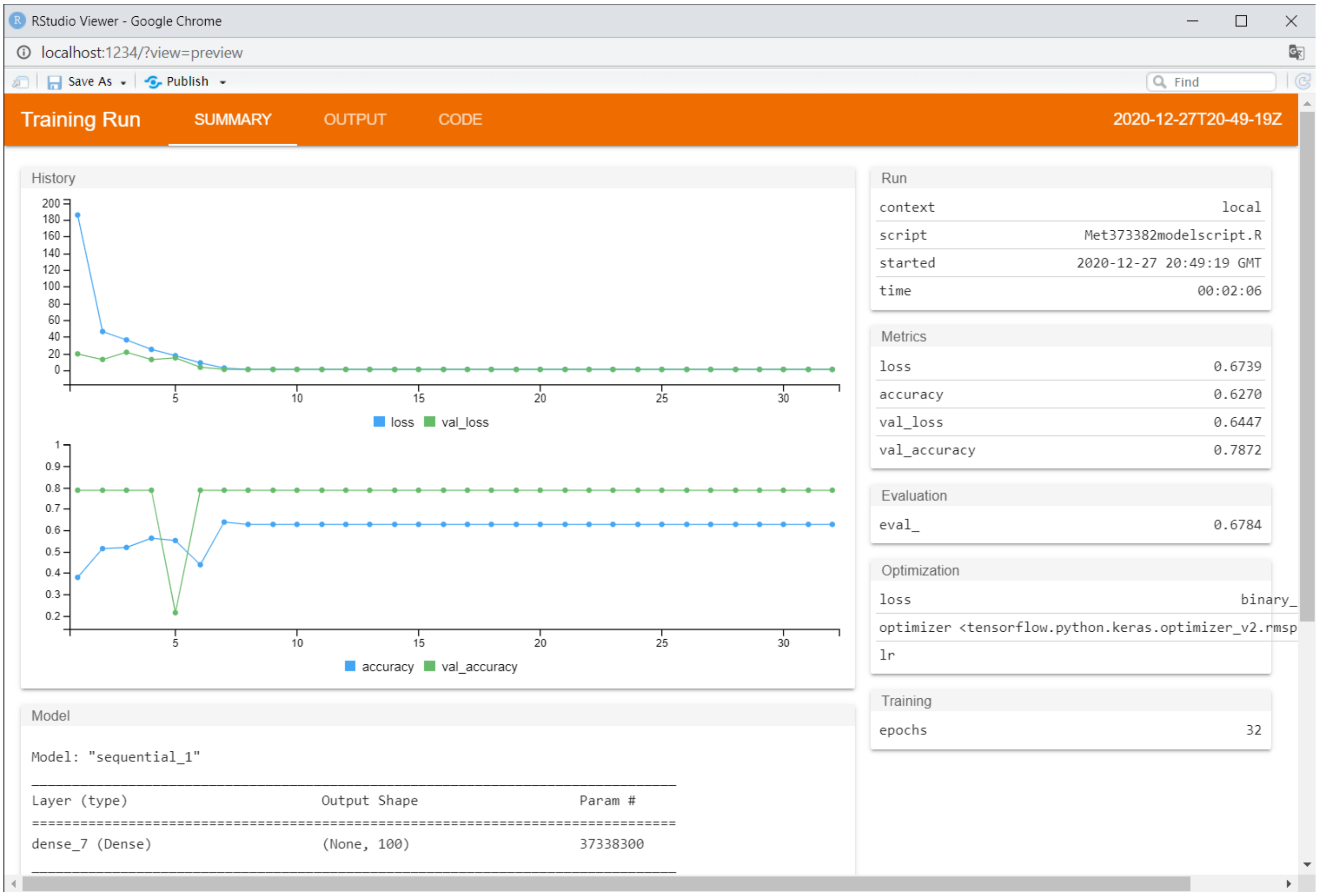
Se obtiene una mayor precisión en el conjunto de validación que en el de entrenamiento. No se alcanza el overfitting. No se puede hacer una red más grande porque no hay espacio suficiente.

Para la red:

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed <- keras_model_sequential() %>% layer_dense(units = 100, activation = “relu”, input_shape = c(373382)) %>% layer_dropout(0.2) %>% layer_dense(units = 50, activation = “relu”) %>% layer_dropout(0.2) %>% layer_dense(units = 1, activation = “sigmoid”)

Obtenemos:

```
> score
     loss  accuracy
0.6783810 0.6034483
```



Run

context	local
script	Met373382modelscript.R
started	2020-12-27 20:49:19 GMT
time	00:02:06

Metrics

loss	0.6739
accuracy	0.6270
val_loss	0.6447
val_accuracy	0.7872

Evaluation

eval_	0.6784

Optimization

loss	binary_
optimizer	<tensorflow.python.keras.optimizer_v2.rmsp
lr	

Training

epochs	32

Model

```
Model: "sequential_1"
________________________________________________________________________________
Layer (type)                        Output Shape                    Param #
================================================================================
dense_7 (Dense)                     (None, 100)                     37338300
________________________________________________________________________________
```

Lo que me llama la atención de esta red es que los conjuntos de validación tienen más precisión en la clasificación que los conjuntos de entrenamiento.

Cuando vemos las predicciones:

```
> predict( Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed,MetTCGA_KIRC_
              [,1]
 [1,] 0.5456889
 [2,] 0.5456889
 [3,] 0.5456889
 [4,] 0.5456889
 [5,] 0.5456889
 [6,] 0.5456889
 [7,] 0.5456889
 [8,] 0.5456889
 [9,] 0.5456889
[10,] 0.5456889
[11,] 0.5456889
[12,] 0.5456889
[13,] 0.5456889
[14,] 0.5456889
[15,] 0.5456889
[16,] 0.5456889
[17,] 0.5456889
[18,] 0.5456889
[19,] 0.5456889
```

```
[20,] 0.5456889
[21,] 0.5456889
[22,] 0.5456889
[23,] 0.5456889
[24,] 0.5456889
[25,] 0.5456889
[26,] 0.5456889
[27,] 0.5456889
[28,] 0.5456889
[29,] 0.5456889
[30,] 0.5456889
[31,] 0.5456889
[32,] 0.5456889
[33,] 0.5456889
[34,] 0.5456889
[35,] 0.5456889
[36,] 0.5456889
[37,] 0.5456889
[38,] 0.5456889
[39,] 0.5456889
[40,] 0.5456889
[41,] 0.5456889
[42,] 0.5456889
[43,] 0.5456889
[44,] 0.5456889
[45,] 0.5456889
[46,] 0.5456889
[47,] 0.5456889
[48,] 0.5456889
[49,] 0.5456889
[50,] 0.5456889
[51,] 0.5456889
[52,] 0.5456889
[53,] 0.5456889
[54,] 0.5456889
[55,] 0.5456889
[56,] 0.5456889
[57,] 0.5456889
[58,] 0.5456889
```

Vemos que tenemos el mismo número una y otra vez, tenemos el mismo problema que con los datos de expresión génica.Si probamos con una red más pequeña:

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed <- keras_model_sequential() %>% layer_dense(units = 16, activation = "relu", input_shape = c(373382)) %>% layer_dropout(0.2) %>% layer_dense(units = 8, activation = "relu") %>% layer_dropout(0.2) %>% layer_dense(units = 1, activation = "sigmoid")

```
> predict( Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProt_Renamed,MetTCGA_KIRC
            [,1]
 [1,] 0.5419121
 [2,] 0.5419121
 [3,] 0.5419121
 [4,] 0.5419121
 [5,] 0.5419121
 [6,] 0.5419121
 [7,] 0.5419121
 [8,] 0.5419121
 [9,] 0.5419121
[10,] 0.5419121
[11,] 0.5419121
[12,] 0.5419121
[13,] 0.5419121
[14,] 0.5419121
[15,] 0.5419121
[16,] 0.5419121
[17,] 0.5419121
[18,] 0.5419121
[19,] 0.5419121
[20,] 0.5419121
[21,] 0.5419121
[22,] 0.5419121
[23,] 0.5419121
[24,] 0.5419121
[25,] 0.5419121
[26,] 0.5419121
[27,] 0.5419121
[28,] 0.5419121
[29,] 0.5419121
[30,] 0.5419121
[31,] 0.5419121
[32,] 0.5419121
[33,] 0.5419121
[34,] 0.5419121
[35,] 0.5419121
[36,] 0.5419121
[37,] 0.5419121
[38,] 0.5419121
[39,] 0.5419121
[40,] 0.5419121
[41,] 0.5419121
[42,] 0.5419121
[43,] 0.5419121
[44,] 0.5419121
[45,] 0.5419121
[46,] 0.5419121
[47,] 0.5419121
[48,] 0.5419121
[49,] 0.5419121
[50,] 0.5419121
[51,] 0.5419121
[52,] 0.5419121
[53,] 0.5419121
[54,] 0.5419121
[55,] 0.5419121
[56,] 0.5419121
[57,] 0.5419121
[58,] 0.5419121
```

```
> history

Final epoch (plot to see history):
        loss: 0.6748
    accuracy: 0.627
    val_loss: 0.6484
val_accuracy: 0.7872
```

Modelos integrados

Lo primero que tenemos que hacer es encontrar el mayor número de muestras con solo dos ómicas. Sabemos las ómicas por separado tendrían este número de muestras:

- Expresión génica (n=606)
- Expresión proteica (n=478)
- Metilación (n=483)

Y sabemos que las ómicas integradas:

- Expresión Génica + Expresión proteica -> n=474
- Expresión Génica + Expresión proteica + Metilación -> n=290

Nos falta por saber cuántas muestras habrá si solo utilizamos:

- Expresión Génica + Metilación
- Expresión Proteica + Metilación

Esto es interesante dado que hemos visto que una ómica no parece ser suficiente para tener una precisión buena para el modelo de clasificación. Vamos a comprobar si al añadirle ómicas la precisión aumenta y, claro está, es cuántas más muestras podamos utilizar para realizar el modelo, mejor.

¿Cuántas muestras podremos utilizar con Expresión Génica + Metilación?

```
# Obtenemos los índices en el vector de muestras de Expresión génica para todas las muestras de metilac

Index_Met_Samples_IN_ExpGen <- c()
for (i in MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA$sample){
  Index_Met_Samples_IN_ExpGen <- c(Index_Met_Samples_IN_ExpGen, which(ExpGenTCGA_KIRC_RawData$sample %i
}
> length(Index_Met_Samples_IN_ExpGen)
[1] 343
```

El vector `Index_Met_Samples_IN_ExpGen` tiene 343 elementos y el vector `MetTCGA_KIRC_RawData_woDupSamples_woSNP_woX` tiene 483 elementos, por lo que hay 100 muestras de la base de datos de metilación que no se encuentran en el dataset de Expresión Génica. ¿Qué muestrasde `MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA$sample` no se encuentran en `ExpGenTCGA_KIRC_RawData`?

```
# Muestras de Metilación (índice numérico) que no están en Expresión Génica
 MetSampleNumb_notIN_ExpGen <-  which(match(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA$sample, E
> MetSampleNumb_notIN_ExpGen
```

```
  [1]   7  10  14  16  20  34  38  40  41  47  57  58  67  69  74  81  85  90  94  96  97  98 100 101
 [25] 103 104 110 111 113 117 123 124 126 127 128 131 133 138 139 140 145 147 155 157 159 166 170 172
 [49] 173 178 185 187 192 198 201 206 208 211 213 221 224 227 231 233 235 250 254 257 259 262 265 267
 [73] 271 279 281 282 286 288 289 291 299 301 302 303 307 308 309 312 313 315 321 324 331 332 333 336
 [97] 337 340 344 347 350 353 360 362 371 373 375 379 381 385 390 391 393 398 401 404 405 408 411 412
[121] 414 420 430 433 434 435 437 438 440 441 445 448 450 451 452 467 470 471 478 479
> ExpGenTCGA_KIRC_RawData$sample[MetSampleNumb_notIN_ExpGen]
```

```
# Quitamos las muestras de Expresión Proteica que no coinciden con muestras de Expresión Génica de los d
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpGen <- MetTCGA_KIRC_RawData_woDupSamples_wo
```

Ahora en metilación tenemos 343 muestras, pero todas esas muestras están en expresión génica. Ahora tenemos que hacer que todas las muestras de Expresión génica están en metilación.

- ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampMet
- ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampMet

```
 ExpGenSampleNumb_notIN_Met <- which(match(ExpGenTCGA_KIRC_RawData$sample, MetTCGA_KIRC_RawData_woDupSam
 > length(ExpGenSampleNumb_notIN_Met)
[1] 263
```

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampMet <- ExpGenTCGA_KIRC_Norm_Trans_Filt75[,-ExpGenSampleNumb_no
> dim(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampMet)
[1] 4897  343

ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampMet <- ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG[,-ExpGe
> dim(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampMet)
[1] 238 343
```

El máximo número de muestras si trabajamos con los datos de Expresión Génica y metilación es de 343.

¿Cuántas muestras podremos utilizar con Expresión Proteica + Metilación?

Necesitamos: `ExpProtTCGA_KIRC_RawData_woNA_ColNamesShort` y `MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_wo`

```
# Muestras de metilación que están en Expresión proteica

> MetSampleNumb_IN_ExpProt_woSameExpGen <- which(match(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA
> length(MetSampleNumb_IN_ExpProt_woSameExpGen)
[1] 291
```

```
# Muestras de Expresión proteica que están en metilación

ExpProtSampleNumb_IN_Met_woSameExpGen <- which(match(ExpProtTCGA_KIRC_RawData_woNA_ColNamesShort, MetTCG
> length(ExpProtSampleNumb_IN_Met_woSameExpGen)
[1] 291
```

Vemos que los datasets comparten 291 muestras. Vamos a obtener estas muestras de cada uno:

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProtwoSameExpGen <- MetTCGA_KIRC_RawData_wo
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_SameSampExpProtwoSameExpGen)
[1] 373382    291

ExpProtTCGA_KIRC_RawData_woNA_SameSampMet_woSameExpGen <- ExpProtTCGA_KIRC_RawData_woNA[,ExpProtSampleNu
> dim(ExpProtTCGA_KIRC_RawData_woNA_SameSampMet_woSameExpGen)
[1] 177 291
```

El máximo número de muestras si trabajamos con los datos de Expresión proteica y metilación es de 291.

Conclusión

Llegamos a la conclusión que para hacer un modelo utilizando solo dos ómicas y que tenga el mayor número de muestras debemos utilizar los datos de expresión génica y de expresión proteica (n=474).

Modelos integrado con 2 omicas (Transcriptómica y proteómica)

Siguiendo otros ejemplos encontrados en bibliografía y en la web. Por ejemplo:

https://www.biorxiv.org/content/10.1101/114892v2.full

Lo mejor sería realizar una integración de las ómicas utilizando autoencoders. Lo que tenemos que hacer es juntar los dos datasets de ómicas, pero para juntarlos tenemos que organizar las filas (muestras/pacientes) de los objetos `ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Renamed_Transposed` o `Creación set train y test para ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed_Transposed` y `ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed`. Por lo que tendremos que crear de nuevos los conjuntos de test y train para estos tras el reordenamiento.

1) Reordenamos datsets por fila
2) Juntamos los datasets (Modelo 1:ExpGen238+ExpProt y Modelo 2:ExpGen4897+ExpProt)
3) Sampling

Reordenamiento de filas

Queremos saber el índice que ocupan las muestras del vector con las muestras ordenadas en el orden del array.

Obtenemos el nombre de las columnas ordenado para `ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed`

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed <- ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampl

colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed) <- ExpGenTCGA_KIRC_Norm_Trans_Filt7

ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed_Transposed <- t(ExpGenTCGA_KIRC_Norm_Trans_Fil

> Indexmatch_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed_Transposed_SORT <- match(sort(row
> Indexmatch_ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed_Transposed_SORT
  [1] 394 427 424 473  86 238 128  99 221 462 102  20  71  13 244 412 429 378 337 306 199  87 132 165
 [25] 356 298 275 170 382 430 239 100 333 285 155  79 357 267  22 388 180 273  88 111 380 166  43 469
 [49] 307 120 443 124 127 161 447 355 411   3 213 334   9 347 348 385 386 200 147 190  59 291 297 216
 [73]  63 236 286 417  68 123 246 375 174 328  95 158 134 256 130 182 189 131 106 317 217  29 376  89
 [97] 248 402  26 152 228 136 242 470 259 287 301  25 346 234  78 194 345 223 260  32 339  62  51 389
```

```
[121]  23 296 262 283 295 329 437  16 163  61  73  27   1  21  80 148  58 214  15 440 465 254  28  69
[145] 323 222 416  24 181 438 232 277 384 454 461 332 240  11 319 423  38 315 428 144 330 261 133 289
[169] 472 226 281   4 426 391 212  48 210 471  49 107 185 358 197 290 434 249 241 336 156   7  82  92
[193] 392   6 191 253  81 467 366 404  66 178  46 324 359 292 367 175 231 342 310  31 353 436 159 230
[217] 183  39 173 151  84 349 126 263  85 252 441 338 403 314 122 145 146  67 218 299   8 439 395  19
[241] 167  47 364 193 344 138 153 255 407 464 381 141 140 203 171  57 104 309 272 374  44 282 302 250
[265] 372 354 187 421 460 352  35 207 431 300 409 258 121 322  42 453 274 204  56 164 327 308 208  93
[289] 406 184 458 422 157  98 455  54   5 202  90 326 196 468 101 444  94 125 288 459 425  33 410  18
[313] 318  34 154  72 316  74 225 312 270 117 219 448 276 137 418  64 278 110 400 220 235 363 229 313
[337] 169 397 370 405 474  12 449 371 442 109 373 176 415 160  65 284 103 265 466 311 331 279 211 192
[361] 377 419 452 269  70 139  53 188 399 457 293 186 379 305 201 224 150 350  10 343 325 383  55 393
[385] 433 463 361 179 105 227 360 162 247 396 368 233 390 116 168 243 257 387  40  50 209 108 320 118
[409] 271 420 445 304 205  36  96 362  60   2  91 119  30 280  14 251 398  75 177 401 294  77 135 237
[433]  17 172  52  41 413 268 215 414 335  37 143 129 195 149 264 351 266 115 112 408  76  83  45  97
[457] 432 456 340 303 113 245 321 114 198 341 206 450 451 369 435 365 446 142
```

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_SameSampExpProt_Renamed_Transposed_RowSort <- ExpGenTCGA_KIRC_Norm_Tra
```

Obtenemos el nombre de las columnas ordenado para `ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Ren`

```
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Renamed <- ExpGenTCGA_KIRC_Norm_Trans_Filt75_23

colnames(ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Renamed) <- ExpGenTCGA_KIRC_Norm_Trans

ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Renamed_Transposed <- t(ExpGenTCGA_KIRC_Norm_Tr

ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_SameSampExpProt_Renamed_Transposed_RowSort <- ExpGenTCGA_KIRC_
```

```
> match(rownames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort), rownames(Exp
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24
 [25]  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48
 [49]  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96
 [97]  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
[121] 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
[145] 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
[169] 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192
[193] 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
[217] 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
[241] 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264
[265] 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288
[289] 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312
[313] 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336
[337] 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
[361] 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384
[385] 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408
[409] 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432
[433] 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456
[457] 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474

> which(match(rownames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort), rowname
integer(0)
```

Obtenemos el nombre de las columnas ordenado para `ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transpo`

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed <- ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen
```

```
colnames(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed) <- ExpProtTCGA_KIRC_RawData_woNA_SameSamp
```

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed <- t(ExpProtTCGA_KIRC_RawData_woNA_SameS
```

```
> Indexmatch_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_SORT <- match(sort(rownames
> Indexmatch_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_SORT
  [1] 155  58 396  80 127 223   1 104  31  70 116   7 444  51 102 439 334  14  12 187 309 348  68 150
 [25] 146 418 429 364 292 179 211 215 433 235 298 253 160 185 196  22  15 407  63 345  74  25 109 159
 [49] 101  26 244 285 367 281  90  27 204 380 436 347 343  41  69  97 100 227 197 130 258 362 117 409
 [73] 349 190 351 213 386 178 296  73 221  23 471  83 189  98 147 217  38 257 278 222 168 283 152 305
 [97] 315 259 473 300 460 360 422  34 134  76 392  93 405 151  32 350 332  85 372 330 126 365  89 264
[121] 233 153 398 103 286  45 140 198 316 400 149 452 306 354 416 402 287 463 450  88   6 357 451 314
[145] 355 339 318 209 183 175 273 275  64 358 218 428  65  62 467 266 417 458 393 361 261 230 174 468
[169]  78 123  57 454  11 399 472 328  59  28  67 470 111 313  16 137 302 105 184 112 279 188 301   4
[193]  53 135 274 445 389 203 237  75 156  87 236 132 366 255 442  86  35 214 441   9  77 207  81 371
[217]   5 282 317 408 414 456 169 138 277 247  72 224 232 459 466  18 167 228 395 449 378 455 423 124
[241] 133 157 359 145 383 142 381 319 120 325 340 125 338 180 346 172  96 295 269 212 391 148 226 195
[265] 271 299 136 246 119 435  42 267   2 234 462 326  49 333 162 201 239 465 254 163 440 122 320 329
[289] 424 432  20 413 139 341 464 446  66 263 401 202 249 430 438 240 321 368 118 250 453 128 260 205
[313] 229 379 356 335 158  79 290 265 369  36  84 252 419 106 310 186  56  43 322 307  94 225 336 420
[337] 437 324 397  37 173 143 403 297 270 289 370 216 276   3  33 193 337 192 394 294  55 447 376 323
[361] 210  44  39 377 311 461 154 200  13 177 176  30 242 412 427 110 238 245 108 443 308 256  24 384
[385] 415  10 406 421 113 268  99 144 272 288 410 304 327 426 469  95 344 171   8 129 457 280 448 291
[409] 231 382 363 206  48 404  54  71 425  60  52  19  21 166 170 248 387 312 434 191  50 165 251 220
[433] 121  47 411  91 262 293 474 182 141 284 353  61 115  17 164 243 194 199  40 181 331 388 373 390
[457] 114 241 107 131 208 374 352  82  46 161 303 375 342  29 385  92 219 431
```

```
ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort <- ExpProtTCGA_KIRC_RawData_woNA
```

Juntar los datasets

```
> ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm_Trans
> dim(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm_T
[1] 474 415
```

```
> ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm_Trans
> dim(ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm_T
[1]  474 5074
```

Sampling: Creación de conjuntos de Test y Train

Modelo integrado con 3 ómicas (Transcriptómica, proteómica y epigenómica)