Informe 10: Modelos integrados

Carmen Lebrero Cia

Contents

Introducción

En el informe anterior vimos que los modelos de Expresión génica y metilación no lograban el overfitting. Se piensa que podría ser un problema referido al rango de datos que encontramos dentro de los datasets, por lo que vamos a volver a normalizar los datos crudos de estas dos ómicas.

(x - min(col))/(max(col)-min(col))

donde x es el dato de una casilla del array y col es la columna de genes o sondas.

Tenemos que retomar los objetos de R:

- ExpGenTCGA_KIRC_Norm : Objeto de Expresión Génica antes de realizar el análisis diferencial. En este objeto utilizamos solo la función `TCGAanalyze_Normalization` que vimos que solo quitaba los decimales.
- MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA
- ExpProtTCGA_KIRC_RawData_woNA

En este caso aún no hemos traspuesto los arrays por lo que las muestras se encuentran en las columnas y aún no hemos modificado el nombre de estas para que sean los nombres cortos.

Tenemos que:

1) Normalizar los datos correctamente
2) Hacer Filtrado no específico al 75%. Se puede hacer con la función `TCGAanalyze_Filtering`
3) Volver a hacer el PCA de los datos e intentar obtener las componentes principales
4) Hacer Análisis de Expresión diferencial y Análisis de metilación diferencial
5) Renombrar datasets y quedarnos con las muestras compartidas entre ómicas
6) Modelos de las ómicas independientes y ver si se comportan de otra manera (si memorizan o aprenden)
7) Modelo de ómicas integradas

1. Normalización de los datasets

Expresión génica

```
> dim(ExpGenTCGA_KIRC_RawData)
[1] 19662   606
ExpGenTCGA_KIRC_Norm01 <- ExpGenTCGA_KIRC_Norm
```

```
for (i in 1:dim(ExpGenTCGA_KIRC_Norm01)[1]){
min <- min(ExpGenTCGA_KIRC_Norm01[i,])
max <- max(ExpGenTCGA_KIRC_Norm01[i,])
for (j in 1:dim(ExpGenTCGA_KIRC_Norm01)[2]){
ExpGenTCGA_KIRC_Norm01[i,j] <- (ExpGenTCGA_KIRC_Norm01[i,j] - min)/(max - min)
print(c(i,j))
}}
```

Metilación

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA)
[1] 373382    483
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm <- MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_
x <- assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA)
for (i in 1:dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)[1]){
min <- min(x[i,])
max <- max(x[i,])
print(i)
for (j in 1:dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)[2]){
x[i,j] <- (x[i,j] - min)/(max - min)
}}
assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm) <- x
```

Proteomica

```
> dim(ExpProtTCGA_KIRC_RawData_woNA)
[1] 177 478
ExpProtTCGA_KIRC_RawData_woNA_Norm <- ExpProtTCGA_KIRC_RawData_woNA
for (i in 1:dim(ExpProtTCGA_KIRC_RawData_woNA_Norm)[1]){
min <- min(ExpProtTCGA_KIRC_RawData_woNA_Norm[i,])
max <- max(ExpProtTCGA_KIRC_RawData_woNA_Norm[i,])
print(i)
for (j in 1:dim(ExpProtTCGA_KIRC_RawData_woNA_Norm)[2]){
ExpProtTCGA_KIRC_RawData_woNA_Norm[i,j] <- (ExpProtTCGA_KIRC_RawData_woNA_Norm[i,j] - min)/(max - min)
}}
```

2. Filtrado no específico al 75%

Expresión génica

```
ExpGenTCGA_KIRC_Norm01_Filt75 <- TCGAanalyze_Filtering(tabDF = ExpGenTCGA_KIRC_Norm01, method = "quanti
> dim(ExpGenTCGA_KIRC_Norm01)
[1] 19586   606
> dim(ExpGenTCGA_KIRC_Norm01_Filt75)
[1] 4897  606
```

Metilación

75%

```
x <- assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
```

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75 <- TCGAanalyze_Filtering(tabDF = x, metho
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
[1] 373382    483
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75)
[1] 93346   483
```

85%

```
x <- assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85 <- TCGAanalyze_Filtering(tabDF = x, metho
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
[1] 373382    483
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85)
[1] 56008   483
```

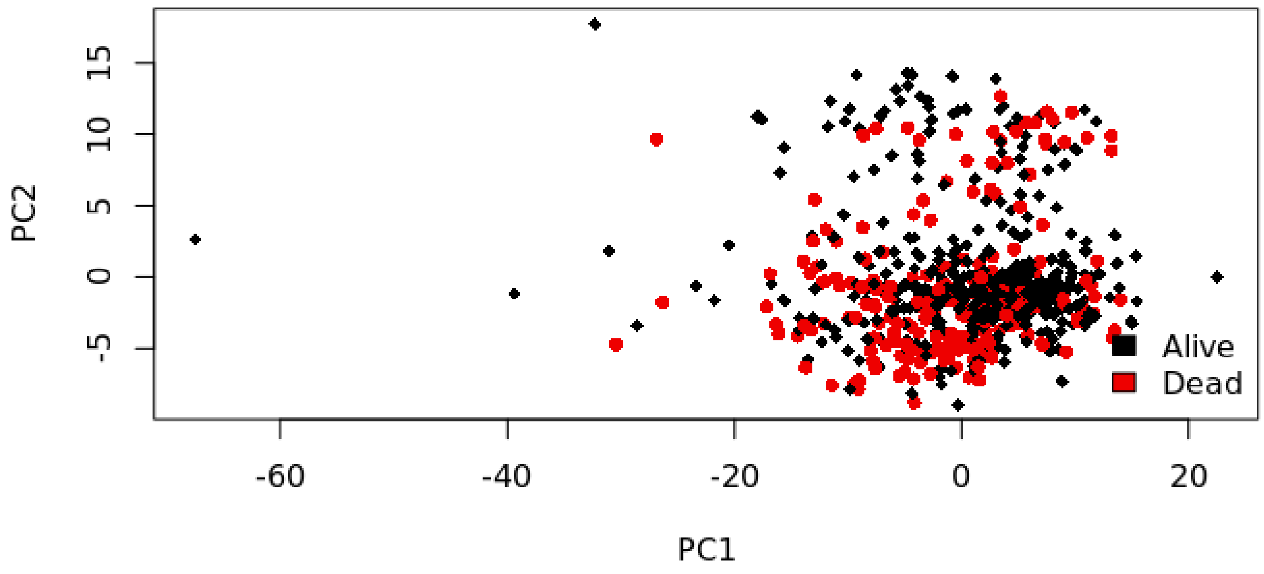
90%

```
x <- assay(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90 <- TCGAanalyze_Filtering(tabDF = x, metho
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)
[1] 373382    483
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90)
[1] 37339   483
```

3. PCA

Expresión génica

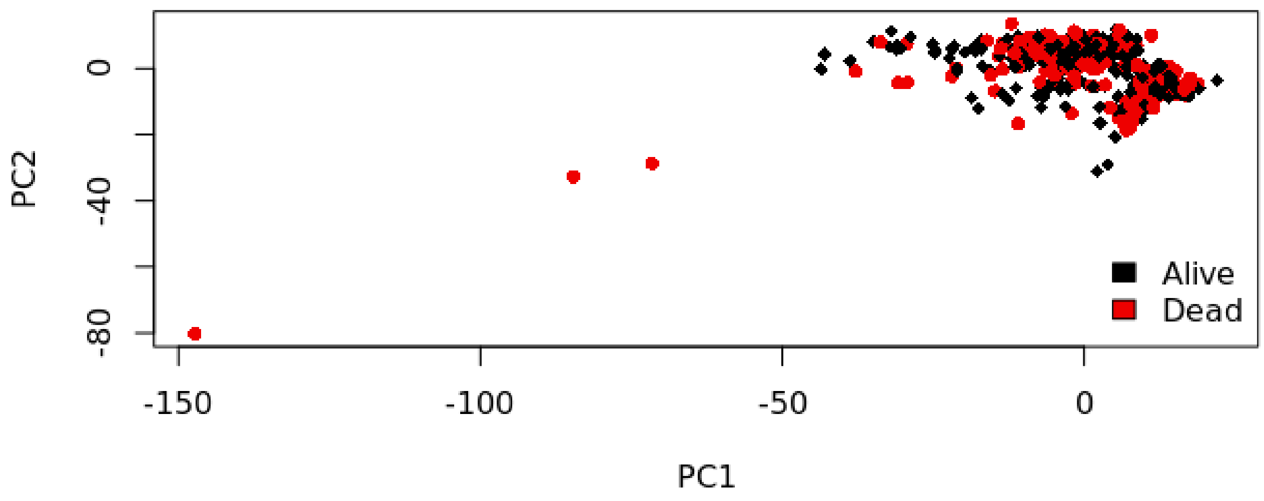
```
ExpGenTCGA_KIRC_Norm01_Filt75
> ExpGenTCGA_KIRC_Norm01_Filt75_PCA <- prcomp(t(ExpGenTCGA_KIRC_Norm01_Filt75))
> str(ExpGenTCGA_KIRC_Norm01_Filt75_PCA)
List of 5
 $ sdev    : num [1:606] 6.98 2.83 2.47 2.02 1.68 ...
 $ rotation: num [1:4897, 1:606] 0.0149 0.0102 0.0125 0.0168 0.0122 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
  .. ..$ : chr [1:606] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 0.311 0.312 0.314 0.342 0.272 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
 $ scale   : logi FALSE
 $ x       : num [1:606, 1:606] -8.884 -1.439 2.618 -0.535 0.183 ...
  ..- 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$vital_status)
> c("black", "red2")[type]
> plot(
ExpGenTCGA_KIRC_Norm_Trans_Filt75_238DEG_Transposed_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



Metilación

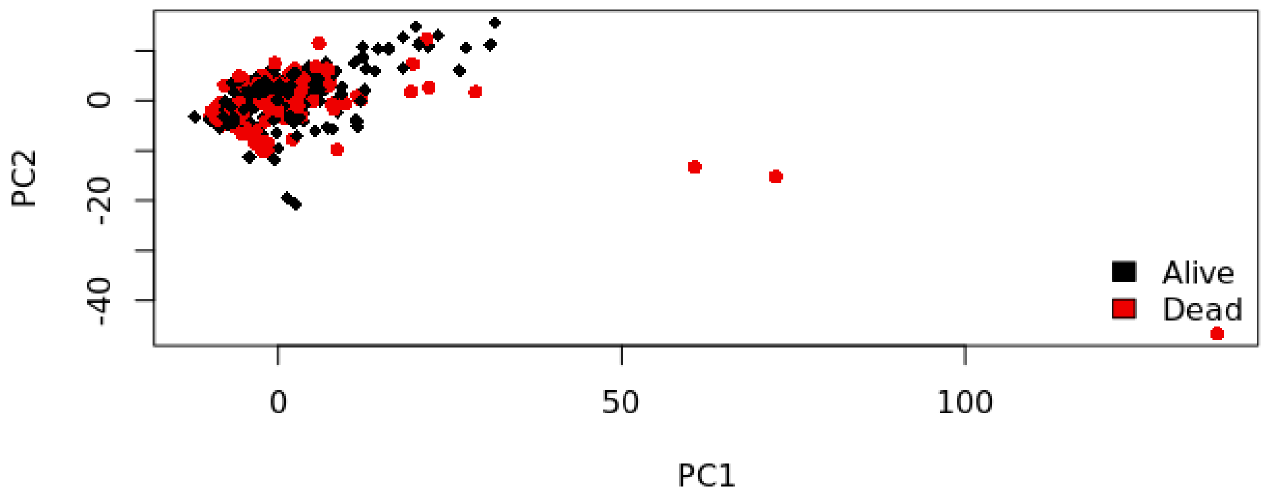
75%

```
#MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75

> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_PCA <- prcomp(t(MetTCGA_KIRC_RawData_wo
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_PCA)
List of 5
 $ sdev    : num [1:483] 14.25 8.19 7.21 6.51 5.48 ...
 $ rotation: num [1:93346, 1:483] 0.000263 0.000311 0.002017 0.000988 0.001926 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
  .. ..$ : chr [1:483] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:93346] 0.887 0.952 0.872 0.841 0.886 ...
  ..- attr(*, "names")= chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
 $ scale   : logi FALSE
 $ x       : num [1:483, 1:483] -0.06359 -21.42776 -3.05577 -0.00718 16.83284 ...
  ..- 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_Norm$vital_status)
> c("black", "red2")[type]
> plot(
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



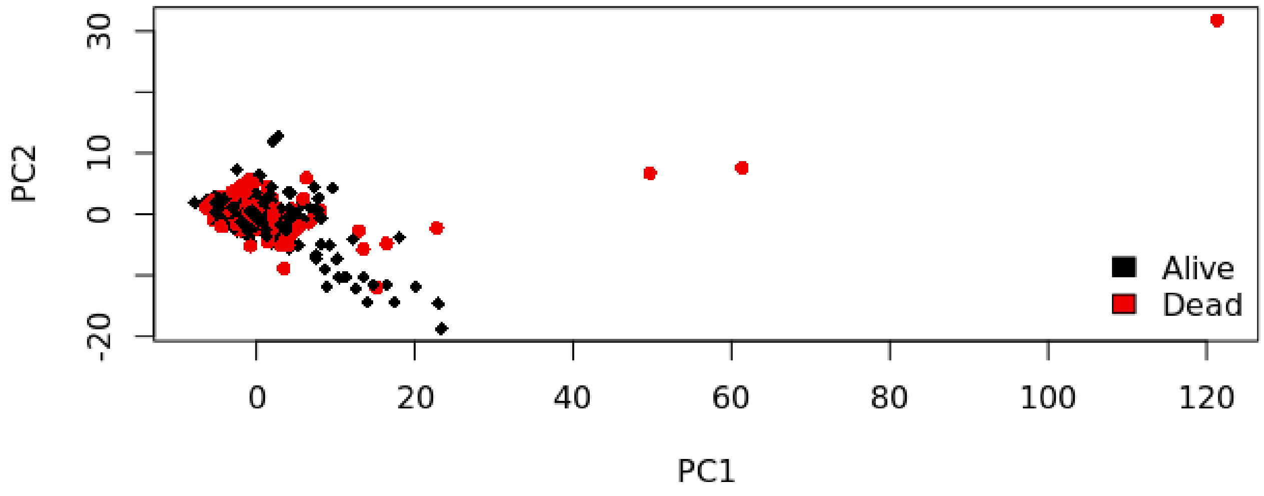
85%

```
#MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85

> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85_PCA <- prcomp(t(MetTCGA_KIRC_RawData_wo
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85_PCA)
List of 5
 $ sdev    : num [1:483] 10.23 5.1 4.41 4.03 3.62 ...
 $ rotation: num [1:56008, 1:483] -0.000417 -0.00052 -0.002611 -0.002566 -0.000308 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:56008] "cg00000236" "cg00000721" "cg00000948" "cg00001364" ...
  .. ..$ : chr [1:483] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:56008] 0.887 0.952 0.872 0.886 0.977 ...
  ..- attr(*, "names")= chr [1:56008] "cg00000236" "cg00000721" "cg00000948" "cg00001364" ...
 $ scale   : logi FALSE
 $ x        : num [1:483, 1:483] -0.917 14.652 0.436 -1.506 -8.847 ...
  ..- 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_Norm$vital_status)
> c("black", "red2")[type]
> plot(
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt85_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



90%

```
#MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90

> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90_PCA <- prcomp(t(MetTCGA_KIRC_RawData_wo
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90_PCA)
List of 5
 $ sdev     : num [1:483] 8.11 3.64 3.06 2.83 2.55 ...
 $ rotation: num [1:37339, 1:483] -0.000662 -0.000262 -0.0066 -0.000648 -0.006593 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:37339] "cg00000721" "cg00001687" "cg00001791" "cg00001854" ...
  .. ..$ : chr [1:483] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:37339] 0.952 0.977 0.903 0.892 0.923 ...
  ..- attr(*, "names")= chr [1:37339] "cg00000721" "cg00001687" "cg00001791" "cg00001854" ...
 $ scale   : logi FALSE
 $ x       : num [1:483, 1:483] -1.01 10.4 -0.19 -1.6 -5.69 ...
  ..- 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_Norm$vital_status)
> c("black", "red2")[type]
> plot(
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt90_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



Proteómica

No puedo hacer un PCA de los datos de proteómica sueltos (solo puedo si cojo las muestras que son iguales que transcriptómica), puesto que no se encuentran metadatos disponibles para descargar acerca de esta ómica.

4. Análisis de Expresión diferencial y Análisis de metilación diferencial

Análisis de Expresión diferencial

Normalizado (ExpGenTCGA_KIRC_Norm01)

Si realizamos el análisis sin el prefiltrado, tarda demasiado (intentar más adelante).

```
ExpGenTCGA_KIRC_SampleName_DeadStatus <- subset(ExpGenTCGA_KIRC_RawData$barcode, ExpGenTCGA_KIRC_RawData
ExpGenTCGA_KIRC_SampleName_AliveStatus <- subset(ExpGenTCGA_KIRC_RawData$barcode, ExpGenTCGA_KIRC_RawDat

ExpGenTCGA_KIRC_Norm01_DEGs <- TCGAanalyze_DEA(mat1 = ExpGenTCGA_KIRC_Norm01[,ExpGenTCGA_KIRC_SampleName
```

Normalizado y filtrado (ExpGenTCGA_KIRC_Norm01_Filt75)

Si realizamos un análisis de expresión diferencial de los datos tras haberlos normalizado y filtrado al 75% a valores entre 0-1 vemos que la función `TCGAanalyze_DEA` no encuentra ningún gen diferencialmente expresado entre las condiciones de vivo o muerto.

```
ExpGenTCGA_KIRC_SampleName_DeadStatus <- subset(ExpGenTCGA_KIRC_RawData$barcode, ExpGenTCGA_KIRC_RawData
ExpGenTCGA_KIRC_SampleName_AliveStatus <- subset(ExpGenTCGA_KIRC_RawData$barcode, ExpGenTCGA_KIRC_RawDat

ExpGenTCGA_KIRC_Norm01_Filt75_DEGs <- TCGAanalyze_DEA(mat1 = ExpGenTCGA_KIRC_Norm01_Filt75[,ExpGenTCGA_

> ExpGenTCGA_KIRC_Norm01_Filt75_DEGs
[1] logFC          logCPM         LR             PValue         FDR            start_position
[7] end_position
<0 rows> (or 0-length row.names)
```

Análisis de metilación diferencial

Por otro lado, vamos a realizar el análisis de metilación diferencial de sitios CpG para el objeto de metilacion normalizado y el objeto normalizado y filtrado con una variabilidad del 75%.

Normalizado (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm)

```
Differentially_metylated_analysis <- TCGAanalyze_DMC(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_
groupCol = "vital_status", # a column in the colData matrix
group1 = "Dead", # a type of the disease type column
group2 = "Alive", # a type of the disease column
p.cut = 0.05,
plot.filename = "survival_metvolcano_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm.png",
diffmean.cut = 0.15,
save = FALSE,
legend = "State",
cores = 1 # if set to 1 there will be a progress bar
 )
```

Normalizado y filtrado (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt

```
Differentially_metylated_analysis <- TCGAanalyze_DMC(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_
groupCol = "vital_status", # a column in the colData matrix
group1 = "Dead", # a type of the disease type column
group2 = "Alive", # a type of the disease column
p.cut = 0.05,
plot.filename = "survival_metvolcano_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75.png"
diffmean.cut = 0.15,
save = FALSE,
legend = "State",
cores = 1 # if set to 1 there will be a progress bar
 )
```

5. Renombrar datasets y muestras compartidas

Expresión génica

Renombrar

```
ExpGenTCGA_KIRC_Norm01_Filt75_ColnamesShort <- c()
for (j in colnames(ExpGenTCGA_KIRC_Norm01_Filt75)){
ExpGenTCGA_KIRC_Norm01_Filt75_ColnamesShort <- c(ExpGenTCGA_KIRC_Norm01_Filt75_ColnamesShort, sub("(.*-
}

ExpGenTCGA_KIRC_Norm01_Filt75_Renamed <- ExpGenTCGA_KIRC_Norm01_Filt75
colnames(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed) <- ExpGenTCGA_KIRC_Norm01_Filt75_ColnamesShort
```

Muestras compartidas

Cuatro objetos distintos vamos a crear:

ExpGenTCGA_KIRC_Norm01_Filt75_Renamed. El objeto de Expresión génica con rango de datos entre 0-1, con variabilidad de genes al 75% pero sin haber realizado un análisis de expresión diferencial.

En este dataset hay 606 muestras entre las que encontramos 404 (66.67%) pacientes vivos y 202 (33.33%) muertos.

```
> dim(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed)
[1] 4897  606
> table(ExpGenTCGA_KIRC_RawData$vital_status)

Alive  Dead
  404   202
```

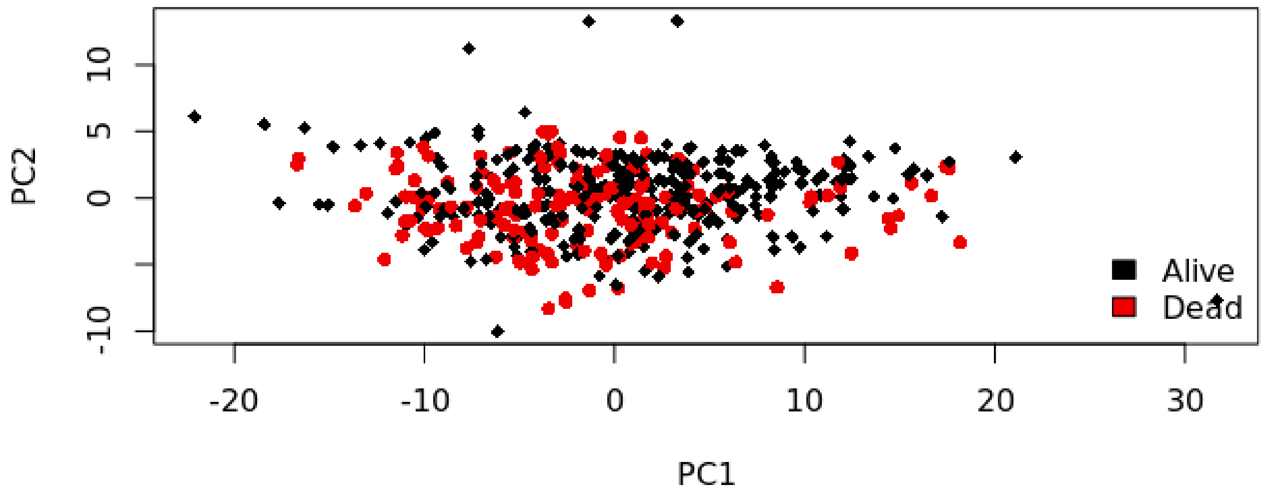
ExpGenTCGA_KIRC_Norm01_Filt75_DEG_Renamed

ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt__Renamed En este dataset hay 474 muestras entre las que encontramos 309 (65.19%) pacientes vivos y 165 (34.81%) muertos.

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed <- ExpGenTCGA_KIRC_Norm01_Filt75_Renamed[,-ExpGen
> dim(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed)
[1] 4897  474

# Labels

x <- c()
for (i in colnames(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed)){
x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
}
> x
  [1]   1   3   4   5   6   7   8  10  13  14  15  16  17  19  20  21  22  23  24  25  26  27  28  29
 [25]  30  31  32  35  36  37  38  40  42  44  45  47  48  50  51  52  53  54  55  56  57  59  60  61
 [49]  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86
 [73]  90  91  92  93  96  97  98  99 100 101 102 103 104 105 106 109 110 112 113 114 116 117 118 119
 [97] 120 121 122 123 124 125 127 128 129 130 132 133 135 136 137 139 143 146 147 148 149 150 151 153
[121] 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 171 173 174 175 176 178 179 180 182
[145] 183 184 185 186 187 189 190 193 195 196 197 198 199 200 201 202 205 206 207 208 209 211 213 216
[169] 217 218 219 220 222 224 227 228 229 230 231 232 233 234 235 236 237 238 239 240 242 244 245 248
[193] 249 250 251 252 254 255 256 257 259 260 261 263 264 265 266 267 268 269 270 272 273 274 275 276
[217] 277 279 280 281 282 283 285 286 289 290 293 294 295 296 299 300 302 303 304 305 307 308 309 311
[241] 312 313 314 315 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 335 336 337 338 339
[265] 340 343 344 345 346 347 348 349 350 351 352 353 354 356 357 358 360 361 362 363 364 365 366 367
[289] 368 369 370 371 372 375 377 378 379 380 381 382 383 384 385 386 387 388 389 390 392 394 395 396
[313] 397 398 401 402 403 405 406 407 408 409 411 413 414 415 417 418 420 421 422 423 424 425 427 428
[337] 429 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 448 449 450 451 452 453 454
[361] 455 456 458 459 460 461 463 464 465 466 467 468 469 470 471 472 475 476 477 478 480 482 483 484
[385] 486 487 488 490 491 492 493 495 496 497 499 500 502 503 504 505 506 507 508 509 510 511 512 514
[409] 515 517 519 520 522 523 525 527 528 529 531 532 533 534 535 537 538 539 540 542 543 545 546 547
[433] 548 549 550 551 553 555 556 557 558 559 560 563 564 565 566 567 568 571 572 573 574 577 579 580
[457] 581 582 583 586 587 591 593 594 595 596 597 598 599 600 601 602 603 605
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels <- ExpGenTCGA_KIRC_RawData$vital_status[x]
> table(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels)
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels
Alive  Dead
  309   165
```

PCA

```
> ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_PCA <- prcomp(t(ExpGenTCGA_KIRC_Norm01_Filt75_S
> str(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_PCA)
List of 5
 $ sdev    : num [1:474] 7.09 2.86 2.32 1.8 1.72 ...
 $ rotation: num [1:4897, 1:474] 0.01676 0.00963 0.01176 0.01563 0.01427 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
  .. ..$ : chr [1:474] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 0.323 0.295 0.309 0.327 0.296 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
 $ scale   : logi FALSE
 $ x       : num [1:474, 1:474] -8.764 2.806 -0.457 0.345 -5.104 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:474] "TCGA-B0-5694-01A" "TCGA-CZ-4860-01A" "TCGA-B0-4706-01A" "TCGA-B4-5844-01A" ...
  .. ..$ : chr [1:474] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels)
> plot(
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



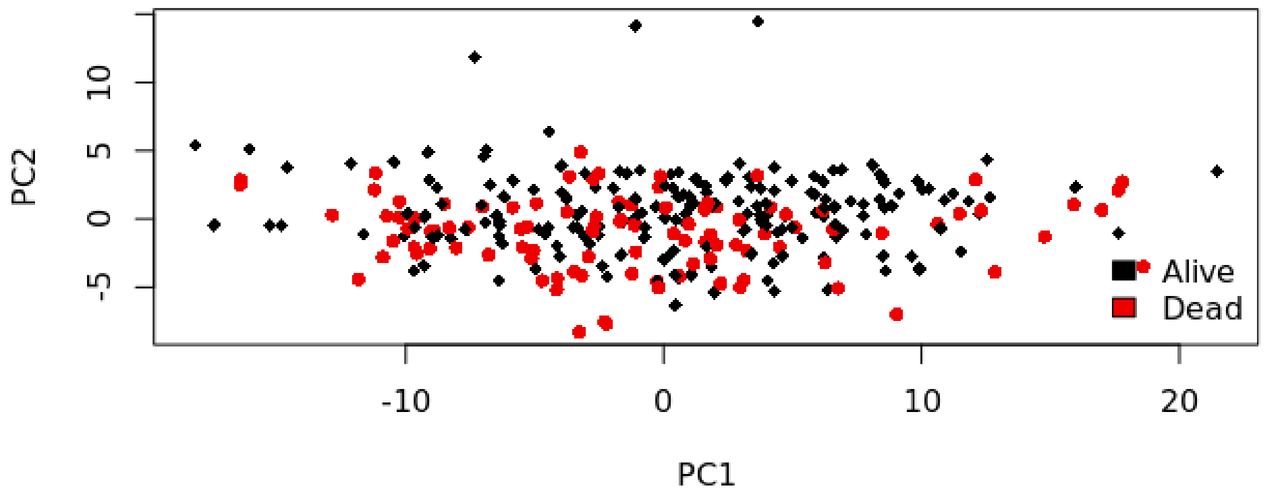
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed En este dataset hay 290 muestras entre las que encontramos 188 (64.83%) pacientes vivos y 102 (35.17%) muertos.

```
ExpGenSampleNumb_IN_Met01 <- which(match(colnames(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed), colnames(MetT

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed <- ExpGenTCGA_KIRC_Norm01_Filt75_Rena

> dim(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed)
[1] 4897  290

# Labels
```

```
x <- c()
for (i in colnames(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed)){
x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
}
> x
  [1]   1   4   5   7   8  10  13  14  15  19  20  21  22  23  26  28  29  30  31  32  35  36  37  40
 [25]  42  44  47  48  50  52  53  55  57  63  64  65  66  67  69  72  73  74  75  76  77  78  80  82
 [49]  83  84  86  90  91  92  93  96  97  99 100 101 102 110 114 117 120 123 124 129 130 132 133 136
 [73] 139 143 146 147 148 149 150 153 156 157 158 160 162 163 164 165 166 168 173 175 179 180 182 185
 [97] 186 187 189 196 198 205 206 207 211 216 218 220 229 231 232 233 234 237 238 240 242 244 245 250
[121] 251 254 255 257 259 265 268 269 272 273 274 275 276 277 280 281 283 285 286 289 290 293 300 302
[145] 303 305 307 312 314 319 320 321 323 325 327 328 330 331 333 335 336 337 339 343 344 345 347 348
[169] 349 352 353 356 358 360 362 365 366 367 368 369 370 372 375 377 378 379 383 385 386 387 389 396
[193] 401 402 403 405 406 407 408 411 414 420 421 423 425 427 428 432 433 434 436 437 438 439 440 441
[217] 443 444 449 451 452 454 455 460 461 464 465 466 472 477 483 484 486 487 488 490 491 492 493 495
[241] 496 497 500 503 504 505 506 509 514 517 519 522 523 525 527 528 529 531 532 535 538 540 542 545
[265] 547 548 549 550 553 555 557 560 563 564 565 566 567 571 572 580 581 582 583 593 595 597 599 600
[289] 601 602
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels <- ExpGenTCGA_KIRC_RawData$vita
> table(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels)
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels
Alive  Dead
  188   102
```

PCA

```
> ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_PCA <- prcomp(t(ExpGenTCGA_KIRC_Norm
> List of 5
 $ sdev    : num [1:290] 7.21 2.96 2.06 1.9 1.8 ...
 $ rotation: num [1:4897, 1:290] 0.01767 0.00995 0.01158 0.01642 0.01411 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:4897] 0.341 0.296 0.315 0.339 0.295 ...
  ..- attr(*, "names")= chr [1:4897] "A2M" "NAT1" "AAMP" "ABCF1" ...
 $ scale   : logi FALSE
 $ x       : num [1:290, 1:290] -8.498 -0.266 0.578 -12.114 6.2 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:290] "TCGA-B0-5694-01A" "TCGA-B0-4706-01A" "TCGA-B4-5844-01A" "TCGA-B8-A54F-01A" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels)
> plot(
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



Metilación

Renombrar

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed <- MetTCGA_KIRC_RawData_woDupSampl

colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed) <- MetTCGA_KIRC_RawData_
```

Muestras compartidas

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed
En este dataset hay 483 muestras entre las que encontramos 299 (61.90%) pacientes vivos y 184 (38.10%) muertos.

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed)
[1] 93346   483
> table(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm$vital_status)

Alive  Dead
  299   184
```

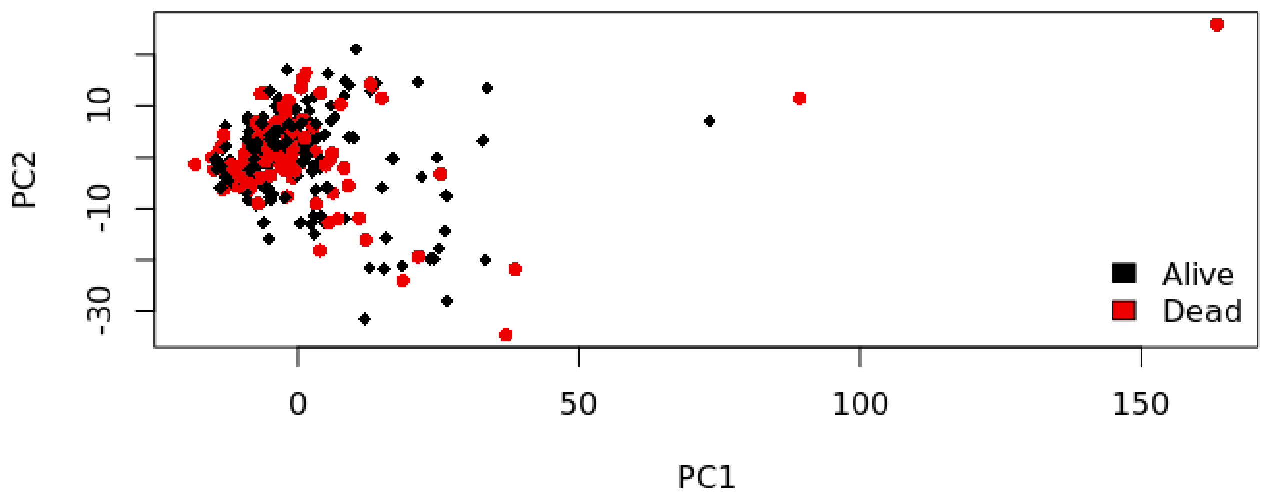
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75___SameSampExpPro
En este dataset hay 291 muestras entre las que encontramos 188 (64.60%) pacientes vivos y 103 (35.39%) muertos.

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed <- MetTCGA_KIRC_
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed)
[1] 93346   291
```

```
# Labels

x <- c()
for (i in colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renam
x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
}
  [1] 582 162 519 372 257 547 503 486 180  29  37 290 339 149 572 385 168 525 265 277 314 272   8  64
 [25] 509 189 560  80 237  15 132 175 555 428 157 396 549 580 280 500 129  52 461 368  69 550 406  86
 [49] 366 377 250 375 369 466 274 436 433 335 557 538 564  47 477 331 234 150 408 303 330 421 302 206
 [73] 300  19 504  21  53 438 244 240 245 465  65  99 545 529 100 407 571  32 460 383 216 505  91 491
 [97]   4 211 532 348 166 231  67 492 323 218 540 437 367 179 187  83 411 273 114  10 102  96 522 336
[121] 403 337 186 595 319 497 379 164 528 207 553 600 449 345 358 182 139 434 427 527 389 535 293 405
[145]  14 439 565 158 602 343 143 123 483 130 327 567  30 153 454  90   5 452 259  74  66  78  36  73
[169] 148 542  76 490 352 356 370 196 160 238 328  77 583 597 268  26 548 563  93  40 378 281 365 444
[193] 440  97 269 325 110  92 493 333 321 402 133 285 484 251 156 472   7 423 320 487  13 289 185 517
[217] 124  84 146  63 163 136 205 455 464 254  48 593 420 275 120 276  57 305 432 401 147  75 599 198
[241] 349 286 229 531  22   1 425 173 283 441  55 353 451 165 344  31  82 496 566 495 117 387 347 386
[265] 443 514 242 362 581 312 101 601  50  44 307  23 232  20 506  28  42 523 255 233 360 414 488 220
[289]  72  35

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels <- ExpGen
>
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels
Alive  Dead
  188   103
```

PCA

```
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_PCA <- prcomp(
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_PCA)
List of 5
 $ sdev    : num [1:291] 15.35 8.46 6.97 6.55 6.41 ...
 $ rotation: num [1:93346, 1:291] -0.000169 -0.000345 -0.001552 -0.001045 -0.001783 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
  .. ..$ : chr [1:291] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:93346] 0.887 0.954 0.857 0.834 0.879 ...
  ..- attr(*, "names")= chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
 $ scale   : logi FALSE
 $ x       : num [1:291, 1:291] -5.23 15.22 -3.19 -6.15 -10.38 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:291] "TCGA-BP-4993-01A" "TCGA-CZ-5982-01A" "TCGA-B0-4703-01A" "TCGA-CJ-4908-01A" ...
  .. ..$ : chr [1:291] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed
> plot(
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_PCA$x,
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



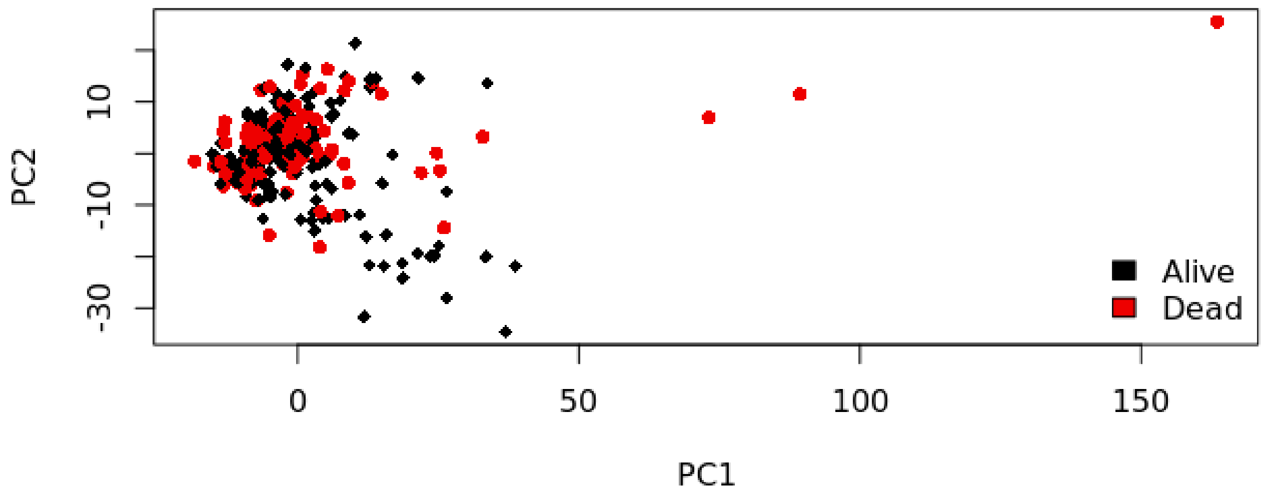
MetTCGA__KIRC__RawData__woDupSamples__woSNP__woXY__woNA__Norm__Filt75__SameSampExpProt
En este dataset hay 290 muestras entre las que encontramos 188 (64.83%) pacientes vivos y 102 (35.17%) muertos.

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

> MetSampleNumb_IN_ExpProt01 <- which(match(colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed <-

> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
[1] 93346   290

# Labels

x <- c()
for (i in colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSa
x <- c(x, which(ExpGenTCGA_KIRC_RawData$sample %in% i))
}
> x
  [1] 582 162 519 372 257 547 503 486 180  29  37 290 339 149 572 385 168 525 265 277 314 272   8  64
 [25] 509 189 560  80 237  15 132 175 555 428 157 396 549 580 280 500 129  52 461 368  69 550 406  86
 [49] 366 377 250 375 369 466 274 436 433 335 557 538 564  47 477 331 234 150 408 303 330 421 302 206
 [73] 300  19 504  21  53 438 244 240 245 465  65  99 545 529 100 407 571  32 460 383 216 505  91 491
 [97]   4 211 532 348 166 231  67 492 323 218 540 437 367 179 187  83 411 273 114  10 102  96 522 336
[121] 403 337 186 595 319 497 379 164 528 207 553 600 449 345 358 182 139 434 427 527 389 535 293 405
```

```
[145]  14 439 565 158 602 343 143 123 483 130 327 567  30 153 454  90   5 452 259  74  66  78  36  73
[169] 148 542  76 490 352 356 370 196 160 238 328  77 583 597 268  26 548 563  93  40 378 281 365 444
[193] 440  97 269 325 110  92 493 333 321 402 133 285 484 251 156 472   7 423 320 487  13 289 185 517
[217] 124  84 146  63 163 136 205 455 464 254  48 593 420 275 120 276  57 305 432 401 147  75 599 198
[241] 349 286 229 531  22   1 425 173 283 441  55 353 451 165 344  31  82 496 566 495 117 387 347 386
[265] 443 514 242 362 581 312 101 601  50  44 307  23 232  20 506  28  42 523 255 233 360 414 488 220
[289]  72  35

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_La
> table(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Re
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_La
Alive  Dead
  188   102
```

PCA

```
> MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_
> str(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Rena
List of 5
 $ sdev    : num [1:290] 15.37 8.46 6.97 6.56 6.42 ...
 $ rotation: num [1:93346, 1:290] -0.000172 -0.000347 -0.001557 -0.001049 -0.001783 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:93346] 0.887 0.954 0.857 0.834 0.879 ...
  ..- attr(*, "names")= chr [1:93346] "cg00000236" "cg00000721" "cg00000948" "cg00001349" ...
 $ scale   : logi FALSE
 $ x       : num [1:290, 1:290] -5.21 15.23 -3.16 -6.13 -10.36 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:290] "TCGA-BP-4993-01A" "TCGA-CZ-5982-01A" "TCGA-B0-4703-01A" "TCGA-CJ-4908-01A" ...
  .. ..$ : chr [1:290] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
> type <- factor(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSamp
> plot(
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_PC
col = c("black", "red2")[type],
pch = c(18, 16)[type],
cex = 1.0)
> legend(
"bottomright",
bty = "n",
c("Alive", "Dead"),
fill = c("black", "red2"),
cex = 1.0)
```



Proteómica

Renombrar

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed <- ExpProtTCGA_KIRC_RawData_woNA_Norm
colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed) <- ExpProtTCGA_KIRC_RawData_woNA_ColNamesShort
```

Muestras compartidas

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed No podemos conocer el porcentaje de vivos y muertos de este dataset puesto que no tenemos las etiquetas (metadatos) de las muestras.

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen En este dataset hay 474 muestras entre las que encontramos 309 (65.19%) pacientes vivos y 165 (34.81%) muertos.

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen <- ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed

> dim(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen)
[1] 177 474

# Labels

x <- c()
for (i in colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen)){
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

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_Labels <- ExpGenTCGA_KIRC_RawData$vital_statu
> table(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_Labels)
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_Labels
Alive  Dead
  309   165
```

ExpProtTCGA__KIRC__RawData__woNA__Norm__Renamed__SameSampMet En este dataset hay 291 muestras entre las que encontramos 188 (64.60%) pacientes vivos y 103 (35.39%) muertos.

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampMet <- ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed[,E

# Labels

x <- c()
for (i in colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampMet)){
x <- c(x, which(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm$sample %in% i))
}
> x
  [1] 195 207  64 384 129 463 382 190   1  15 287  70 168 261 220 457 143 297 328 219  23 449 226 327
 [25] 376 413 386 473 481  78 436 293 475 183 329   2  45 346 372 426 422  37  75 247 163 369  61 345
 [49]  92 144 432 134 158 260  32  11 114 263 358 234 453 180 418  88 241 423 407  44  52  63   8 349
 [73] 212 186 415  17 396 468  36 137  26 339 355  79 218 230 119 374 264 109 354 320 177 210  43 342
 [97] 402 425 229  18 199 223  31 125 477 175  51   4 454 352 150 447 472 474 455 427 275  35 395 465
[121] 248 365 130 399 141 310   3 462 444 249  13 216 251 106 244 460 203   6 240 202 174 466  66  39
[145] 317  60 102 326  42 446 330 306  27 258 252 215 278 366 116 200 165  55 236 239 406 341 120 142
[169]  28 255 403 343 377 276 383  53  76 482  62 363 107 222 357 456  56  30 118 334 416 322 476 151
[193] 368 335 268 191 217 232 439 162 179 157 319 318 136  73 419  25 359 105 417 305  77 325 108  12
[217] 314 193 295 300 225 294  95 182 284 296 246 443  89 392 209 154 196 459 167 169 184 431  71 429
[241]   9 197 152 388 160 283  82 204 338 400 205  99 242 378 389 380 274 442 316 148 458 424 176 273
[265] 280 270 356 290 188 409 194 367  84 256 153 464 483 156  93 266 311 397  50  83  72  68 161  46
[289]  33 428 387

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampMet_Labels <- MetTCGA_KIRC_RawData_woDupSamples_woSN
> table(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampMet_Labels)
```

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampMet_Labels
Alive  Dead
  188   103
```

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet
En este dataset hay 290 muestras entre las que encontramos 188 (64.83%) pacientes vivos y 102 (35.17%) muertos.

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

> ExpProt01SampleNumb_IN_Met <- which(match(colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSamp

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet <- ExpProtTCGA_KIRC_RawData_woNA_

# Labels

x <- c()
for (i in colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet)){
x <- c(x, which(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm$sample %in% i))
}
> x
  [1] 195 207  64 384 129 463 382 190   1  15 287  70 168 261 220 457 143 297 328 219  23 449 226 327
 [25] 376 413 386 473 481  78 436 293 475 183 329   2  45 346 372 426 422  37  75 247 163 369  61 345
 [49]  92 144 432 134 158 260  32  11 114 263 358 234 453 180 418  88 241 423 407  44  52  63   8 349
 [73] 212 186 415  17 396 468  36 137  26 339 355  79 218 230 119 374 264 109 354 320 177 210  43 342
 [97] 402 425 229  18 199 223  31 125 477 175  51   4 454 352 150 447 472 474 455 427 275  35 395 465
[121] 248 365 130 399 141 310   3 462 444 249  13 216 251 106 244 460 203   6 240 202 174 466  66  39
[145] 317  60 102 326  42 446 330 306  27 258 252 215 278 366 116 200 165  55 236 239 406 341 120 142
[169]  28 255 403 343 377 276 383  53  76 482  62 363 107 222 357 456  56  30 118 334 416 322 476 151
[193] 368 335 268 191 217 232 439 162 179 319 318 136  73 419  25 359 105 417 305  77 325 108  12 314
[217] 193 295 300 225 294  95 182 284 296 246 443  89 392 209 154 196 459 167 169 184 431  71 429   9
[241] 197 152 388 160 283  82 204 338 400 205  99 242 378 389 380 274 442 316 148 458 424 176 273 280
[265] 270 356 290 188 409 194 367  84 256 153 464 483 156  93 266 311 397  50  83  72  68 161  46  33
[289] 428 387

ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet_Labels <- MetTCGA_KIRC_RawData_wo
> table(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet_Labels)
ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed_SameSampExpGen_SameSampMet_Labels
Alive  Dead
  188   102
```

6. Modelos independientes de ómicas

Expresión génica

Modelo de Expresión génica (ExpGenTCGA_KIRC_Norm01_Filt75_Renamed: genes = 4897, n = 606)

Usamos: ExpGenTCGA_KIRC_Norm01_Filt75_Renamed

```
> dim(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed)
[1] 4897  606
```

Creación de conjuntos test y train

```
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed <- t(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed)

set.seed(231)

ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Index_Training <- sample(1:nrow(ExpGenTCGA_KIRC_Norm01_
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Test <- ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transpose
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train <- ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transpos
```

Obtención de etiquetas

```
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test <- ExpGenTCGA_KIRC_RawData$vital_status[-ExpGenTCGA_KI
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Train <- ExpGenTCGA_KIRC_RawData$vital_status[ExpGenTCGA_KI

ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test_FactorNumb <- as.integer(factor(ExpGenTCGA_KIRC_Norm01
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test_FactorNumb[ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Label

ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Train_FactorNumb <- as.integer(factor(ExpGenTCGA_KIRC_Norm0
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Train_FactorNumb[ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labe
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed, file = "ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Tr
```

Conjuntos Train y Test

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train, file = "ExpGenTCGA_KIRC_Norm01_Filt75_Rena

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Test, file = "ExpGenTCGA_KIRC_Norm01_Filt75_Renam
```

Etiquetas

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Train_FactorNumb, file = "ExpGenTCGA_KIRC_Norm01_Filt

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test_FactorNumb, file = "ExpGenTCGA_KIRC_Norm01_Filt7
```

Creando script (ExpGenTCGA__KIRC__Norm01__Filt75modelscript.R)

```
# Cargando los archivos

# test_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Test.rda")

# train_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train.rda")

# test_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test_FactorNumb.rda")

# train_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Train_FactorNumb.rda")

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

# Definición del modelo ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed: genes = 4897, n = 606
Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed <- 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_Norm01_Filt75_Renamed_Transposed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed %>% fit(ExpGenTCGA_KIRC_Norm01_Filt75_

plot(history)

score <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed %>% evaluate(
 ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Test, ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Te
  verbose = 0
)

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

Resultados

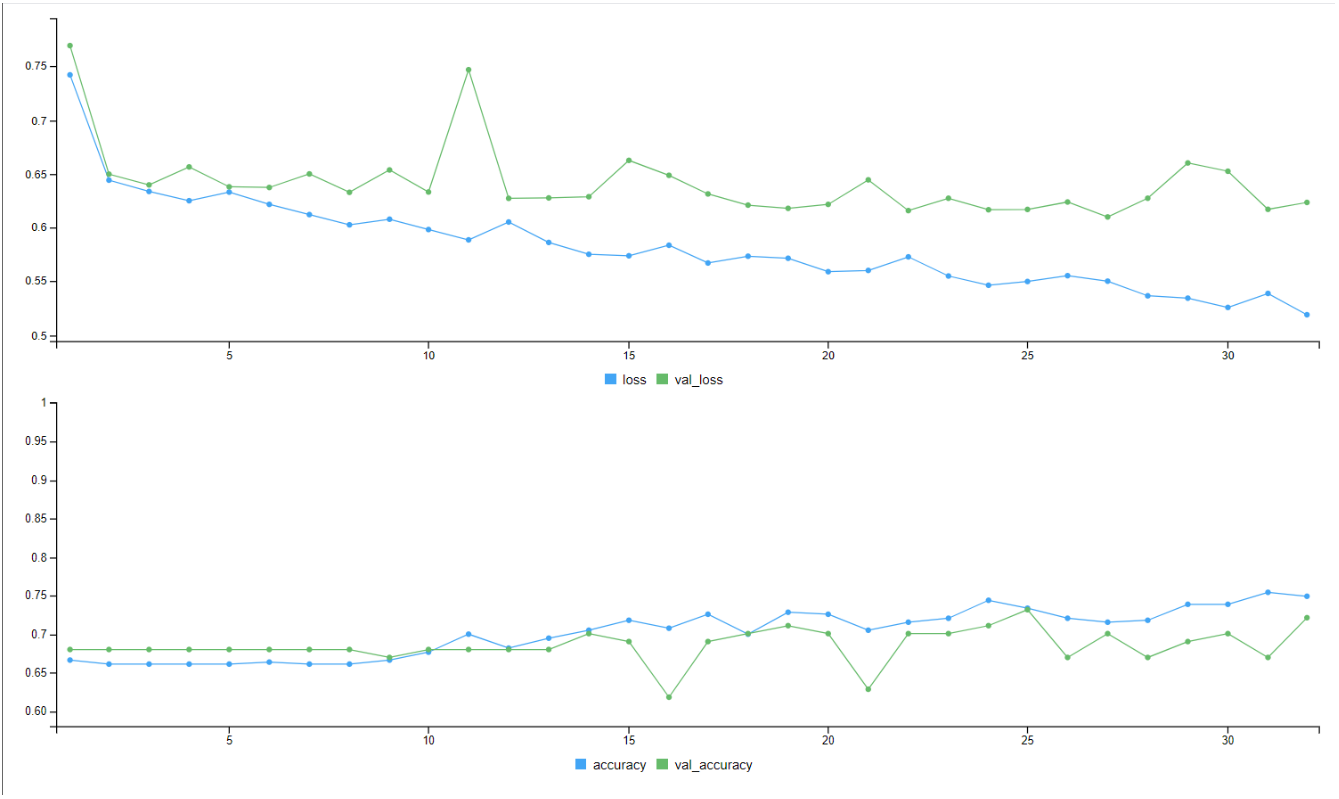
Modelo 1 Resumen del modelo

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed)
```

```
Model: "sequential_2"
____________________________________________________________________________________________________
Layer (type)                              Output Shape                        Param #
====================================================================================================
dense_8 (Dense)                           (None, 16)                          78368
____________________________________________________________________________________________________
dense_7 (Dense)                           (None, 16)                          272
____________________________________________________________________________________________________
dense_6 (Dense)                           (None, 1)                           17
====================================================================================================
Total params: 78,657
Trainable params: 78,657
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.5348775 0.7049180

> history

Final epoch (plot to see history):
        loss: 0.519
    accuracy: 0.7494
    val_loss: 0.6234
val_accuracy: 0.7216
```

```
> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed, ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_
           [,1]
  [1,] 0.8697336
  [2,] 0.8963979
  [3,] 0.7891341
  [4,] 0.9498302
  [5,] 0.8369223
  [6,] 0.6368501
  [7,] 0.8639055
  [8,] 0.6090716
  [9,] 0.7070348
 [10,] 0.9417554
 [11,] 0.8116871
 [12,] 0.6496152
 [13,] 0.5939503
 [14,] 0.9229082
 [15,] 0.8405334
 [16,] 0.5555996
 [17,] 0.8322977
 [18,] 0.6479468
 [19,] 0.5068920
 [20,] 0.5940652
 [21,] 0.5033886
 [22,] 0.9677508
 [23,] 0.9109388
 [24,] 0.9026528
 [25,] 0.5398332
 [26,] 0.6096040
 [27,] 0.7442436
 [28,] 0.4699266
```

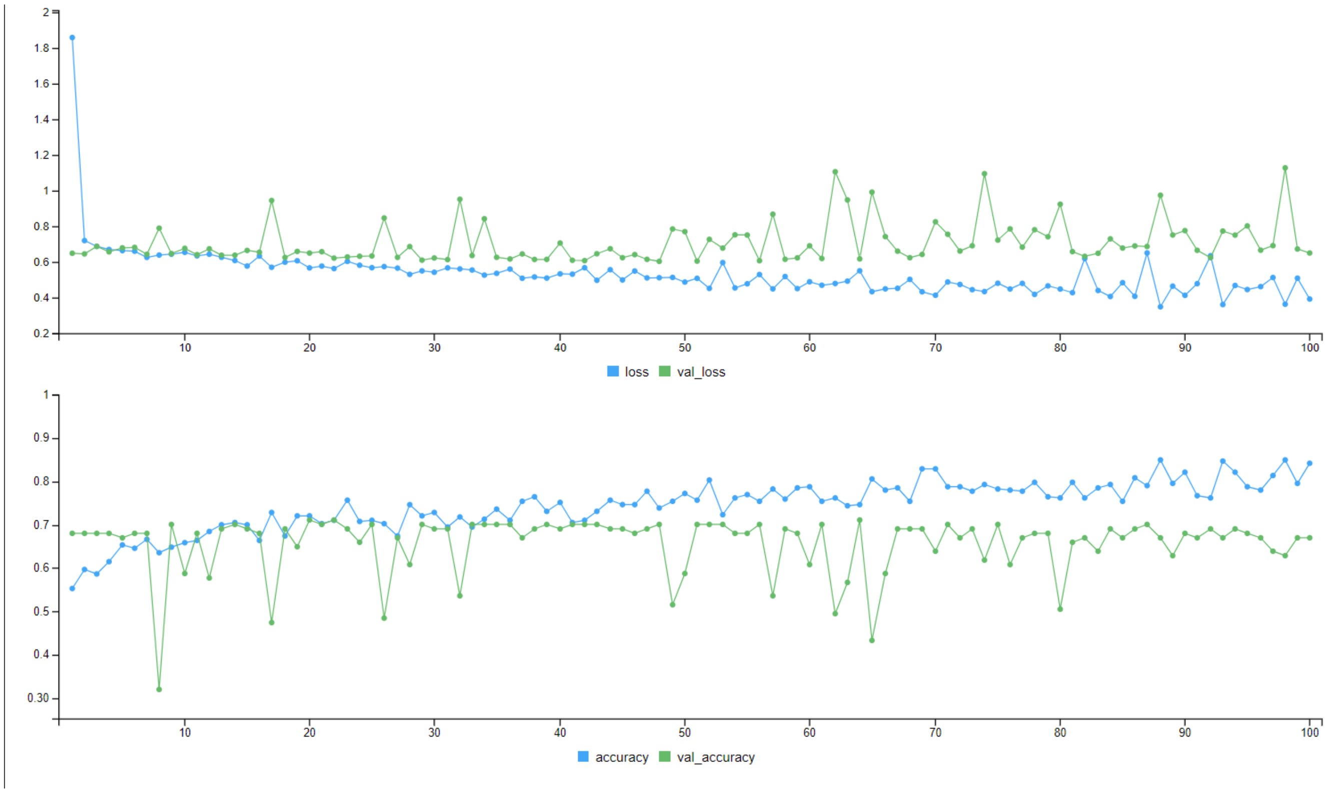
Modelo 2 Resumen del modelo

Vamos a aumentar las epochs de 32 a 100, a ver si encontramos sobreajuste

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed)
Model: "sequential_8"
____________________________________________________________________________________________________
Layer (type)                                  Output Shape                                 Param #
====================================================================================================
dense_30 (Dense)                              (None, 400)                                  1959200
____________________________________________________________________________________________________
dense_29 (Dense)                              (None, 200)                                  80200
____________________________________________________________________________________________________
dense_28 (Dense)                              (None, 100)                                  20100
____________________________________________________________________________________________________
dense_27 (Dense)                              (None, 50)                                   5050
____________________________________________________________________________________________________
dense_26 (Dense)                              (None, 1)                                    51
====================================================================================================
Total params: 2,064,601
Trainable params: 2,064,601
Non-trainable params: 0
____________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.5257909 0.7295082


> history

Final epoch (plot to see history):
        loss: 0.3912
    accuracy: 0.8424
    val_loss: 0.6498
val_accuracy: 0.6701


> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed, ExpGenTCGA_KIRC_Norm01_Filt75_Renamed
             [,1]
  [1,] 0.80715787
  [2,] 0.90325236
  [3,] 0.78039771
  [4,] 0.95047009
  [5,] 0.99955904
```

```
 [6,] 0.55102897
 [7,] 0.85213304
 [8,] 0.67574543
 [9,] 0.57154405
[10,] 0.95356965
[11,] 0.84602123
[12,] 0.73663223
[13,] 0.48719525
[14,] 0.94099689
[15,] 0.68009329
[16,] 0.64478403
[17,] 0.85300636
[18,] 0.57713234
[19,] 0.54804152
[20,] 0.59751326
[21,] 0.48638651
[22,] 0.97435153
[23,] 0.92414105
[24,] 0.92769742
[25,] 0.42746255
[26,] 0.57173383
[27,] 0.77193344
```

Conclusiones A pesar de que no conseguimos overfitting, ahora la red sí que parece aprender un poco más ya que en las predicts encontramos números más cercanos a 1 y más cercanos a 0.

Modelo de Expresión génica (ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed: genes = 4897, n = 474)

Usamos: ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed

```
> dim(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed)
[1] 4897  606
```

Creación de conjuntos test y train

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed <- t(ExpGenTCGA_KIRC_Norm01_Filt75_Same

set.seed(231)

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Index_Training <- sample(1:nrow(ExpGen

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Test <- ExpGenTCGA_KIRC_Norm01_Filt75_

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Train <- ExpGenTCGA_KIRC_Norm01_Filt75
```

Obtención de etiquetas

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Labels_Test <- ExpGenTCGA_KIRC_Norm01_
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Labels_Train <- ExpGenTCGA_KIRC_Norm01
```

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Test_FactorNumb <- as.integer(factor(ExpGe
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Test_FactorNumb[ExpGenTCGA_KIRC_Norm01_Fil

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Train_FactorNumb <- as.integer(factor(ExpG
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Train_FactorNumb[ExpGenTCGA_KIRC_Norm01_Fi
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed, file = "ExpGenTCGA_KIRC_Norm01_Filt75_SameS
```

Conjuntos Train y Test

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Train, file = "ExpGenTCGA_KIRC_No

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Test, file = "ExpGenTCGA_KIRC_Nor
```

Etiquetas

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Train_FactorNumb, file = "ExpGenTCGA_

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Test_FactorNumb, file = "ExpGenTCGA_K
```

Creando script (ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProtmodelscript.R)

```
# Cargando los archivos

# test_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Test.rda")

# train_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Train.rda")

# test_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Test_FactorNumb.rda")

# train_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Train_FactorNumb.rda")

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

```
# Definición del modelo ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed: genes = 4897, n = 474
Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed <- 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_Norm01_Filt75_SameSampExpProt_Renamed_Transposed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed %>% fit(ExpGenTCGA_KI

plot(history)

score <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed %>% evaluate(
 ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Test, ExpGenTCGA_KIRC_Norm01_Filt75_Sa
  verbose = 0
)

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

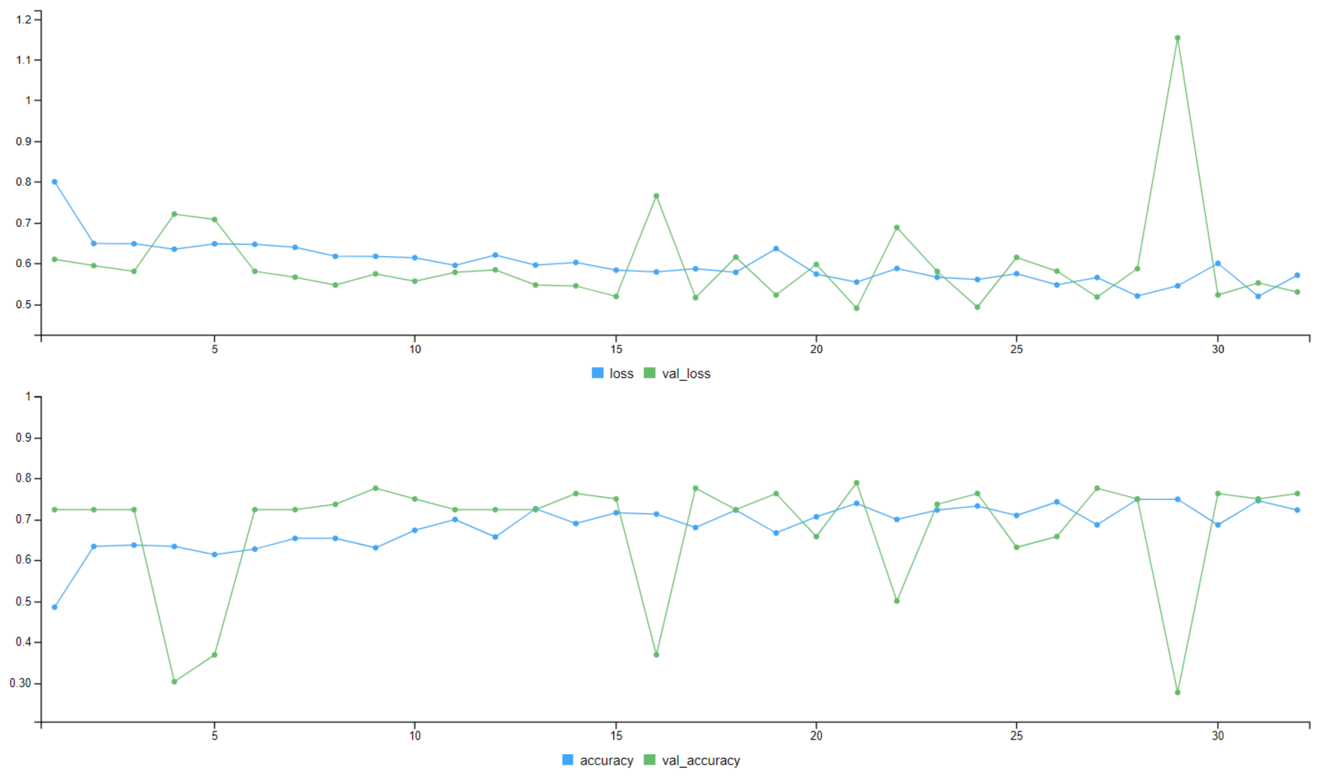
Resultados

Modelo 1 Resumen del modelo

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed)
Model: "sequential_10"
________________________________________________________________________________________________________
Layer (type)                                 Output Shape                                Param #
========================================================================================================
dense_36 (Dense)                             (None, 16)                                  78368
________________________________________________________________________________________________________
dense_35 (Dense)                             (None, 16)                                  272
________________________________________________________________________________________________________
dense_34 (Dense)                             (None, 1)                                   17
========================================================================================================
Total params: 78,657
Trainable params: 78,657
Non-trainable params: 0
________________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.7499498 0.6736842

> history

 history

Final epoch (plot to see history):
        loss: 0.571
    accuracy: 0.7228
    val_loss: 0.5297
val_accuracy: 0.7632

> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed, ExpGenTCGA_KIRC_Norm0
           [,1]
 [1,] 0.9568077
 [2,] 0.8131423
 [3,] 0.9724365
 [4,] 0.8784594
 [5,] 0.9493513
 [6,] 0.6912041
 [7,] 0.9366431
 [8,] 0.9188679
 [9,] 0.6161845
[10,] 0.6595472
[11,] 0.9713345
[12,] 0.7594427
```

```
[13,] 0.8381165
[14,] 0.6564200
[15,] 0.6075458
[16,] 0.9662790
[17,] 0.7718409
[18,] 0.8716229
[19,] 0.9890674
[20,] 0.9213936
[21,] 0.9744039
[22,] 0.8079967
[23,] 0.9582961
[24,] 0.9827141
[25,] 0.2664481
[26,] 0.7576817
[27,] 0.8992621
[28,] 0.4911505
[29,] 0.8541773
[30,] 0.7387527
[31,] 0.5711002
[32,] 0.8454475
[33,] 0.8391758
[34,] 0.8886519
```

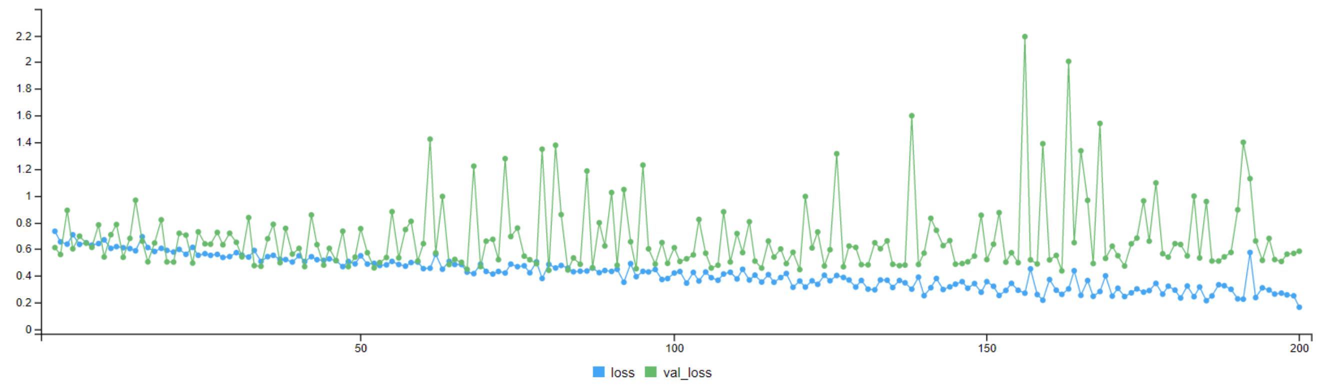
Modelo 2 Resumen del modelo

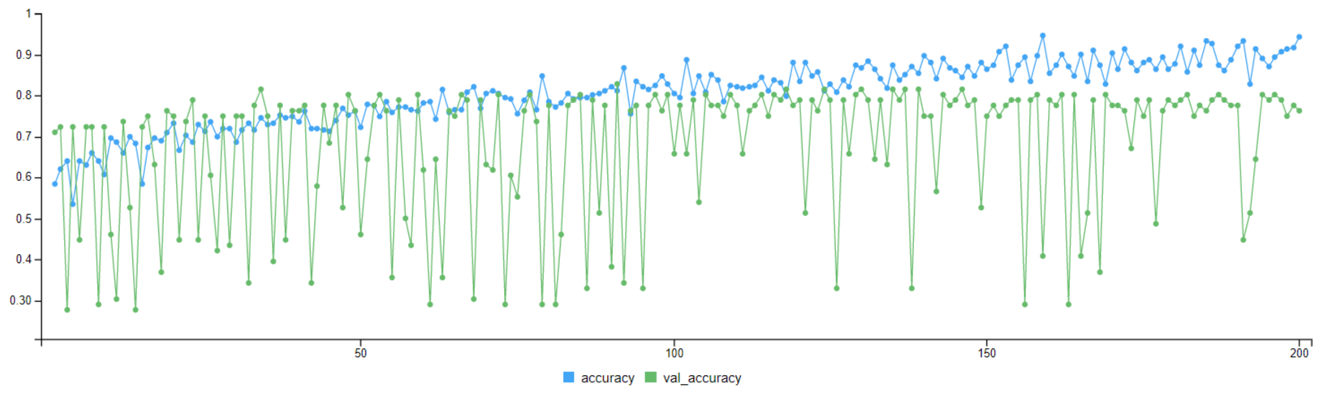
Vamos a aumentar las epochs de 32 a 200. Podemos observar un sobreajuste del modelo, y aunque vemos que la precision en el de validación y de prueba no es mala, en la gráfica de precisión observamos picos extraños en el conjunto de validación.

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed)
Model: "sequential_11"
________________________________________________________________________________________________
Layer (type)                                 Output Shape                            Param #
================================================================================================
dense_39 (Dense)                             (None, 16)                              78368
________________________________________________________________________________________________
dense_38 (Dense)                             (None, 16)                              272
________________________________________________________________________________________________
dense_37 (Dense)                             (None, 1)                               17
================================================================================================
Total params: 78,657
Trainable params: 78,657
Non-trainable params: 0
________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.9051554 0.6631579
> history

Final epoch (plot to see history):
        loss: 0.1643
    accuracy: 0.9439
    val_loss: 0.5839
val_accuracy: 0.7632
> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed, ExpGenTCGA_KIRC_Norm0
              [,1]
 [1,] 9.025037e-01
 [2,] 1.459529e-01
 [3,] 9.691236e-01
 [4,] 9.416673e-01
 [5,] 9.207168e-01
 [6,] 1.361501e-02
 [7,] 1.049048e-01
 [8,] 6.768511e-01
 [9,] 8.850604e-03
[10,] 1.898098e-01
[11,] 9.774320e-01
[12,] 3.574330e-02
[13,] 2.186203e-02
[14,] 6.725825e-01
[15,] 6.669530e-01
[16,] 9.066889e-01
```

```
[17,] 1.798538e-01
[18,] 4.095941e-01
[19,] 9.837865e-01
[20,] 8.323203e-01
[21,] 9.902082e-01
[22,] 7.857132e-01
[23,] 9.524424e-01
[24,] 9.999757e-01
[25,] 5.250251e-05
[26,] 4.366115e-02
[27,] 1.661978e-01
[28,] 2.193150e-02
```

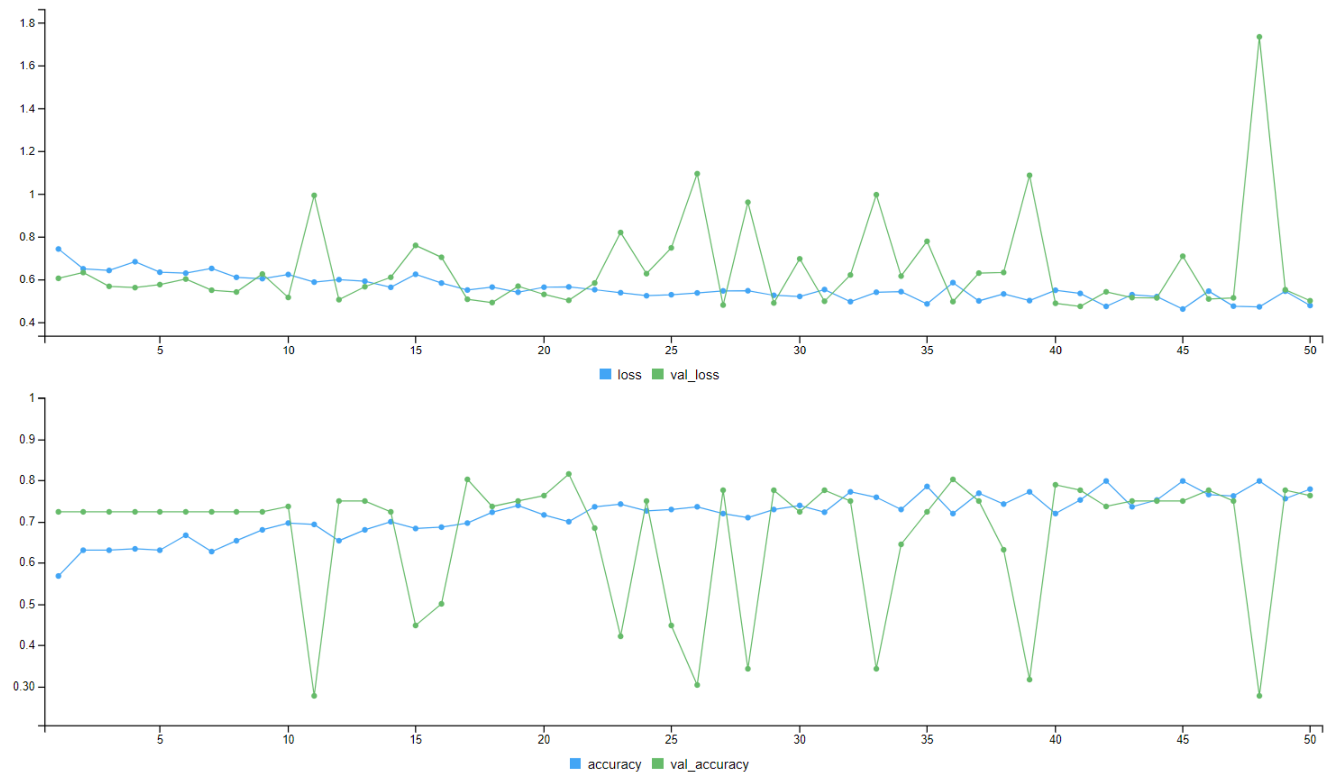
Modelo 3 Resumen del modelo

Hemos aumentado capas y las epochs a 50.

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed)
Model: "sequential_14"
____________________________________________________________________________________________________
Layer (type)                                  Output Shape                              Param #
====================================================================================================
dense_51 (Dense)                              (None, 20)                                97960
____________________________________________________________________________________________________
dense_50 (Dense)                              (None, 16)                                336
____________________________________________________________________________________________________
dense_49 (Dense)                              (None, 8)                                 136
____________________________________________________________________________________________________
dense_48 (Dense)                              (None, 1)                                 9
====================================================================================================
Total params: 98,441
Trainable params: 98,441
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.6106607 0.7263158
> history

Final epoch (plot to see history):
        loss: 0.4787
    accuracy: 0.7789
    val_loss: 0.5008
val_accuracy: 0.7632
> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed, ExpGenTCGA_KIRC_Norm0
            [,1]
 [1,] 0.83467633
 [2,] 0.31380272
 [3,] 0.89783919
 [4,] 0.60204035
 [5,] 0.86398113
 [6,] 0.18751249
 [7,] 0.65454161
 [8,] 0.74137282
 [9,] 0.17497104
[10,] 0.37883353
[11,] 0.85166055
[12,] 0.26118419
[13,] 0.31547427
[14,] 0.46910909
```

Modelo de Expresión génica (ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt__SameSampMet_ genes = 4897, n = 290)

Usamos: ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt__SameSampMet__Renamed

```
> dim(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed)
[1] 4897  290
```

Creación de conjuntos test y train

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed <- t(ExpGenTCGA_KIRC_Norm01

set.seed(231)

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Index_Training <- sample(1

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test <- ExpGenTCGA_KIRC_No

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train <- ExpGenTCGA_KIRC_N
```

Obtención de etiquetas

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Labels_Test <- ExpGenTCGA_

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Labels_Train <- ExpGenTCGA_

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb <- as.integer(
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb[ExpGenTCGA_KIR

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb <- as.integer
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb[ExpGenTCGA_KI
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed, file = "ExpGenTCGA_K
```

Conjuntos Train y Test

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train, file = "ExpGen

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, file = "ExpGenT
```

Etiquetas

```
# Train
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb, file = 

# Test
save(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb, file = "
```

Creando script (ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMetmodelscript.R

```
# Cargando los archivos

# test_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test.rda")

# train_x
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Train.rda")

# test_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Test_FactorNumb.rda")

# train_y
load("ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Labels_Train_FactorNumb.rda")

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

# Definición del modelo ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed: genes = 4897
Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed <- keras_model_seque
  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_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed %>% fit(E

plot(history)

score <- Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed %>% evaluat
 ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed_Test, ExpGenTCGA_KIRC_Nor
  verbose = 0
)

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

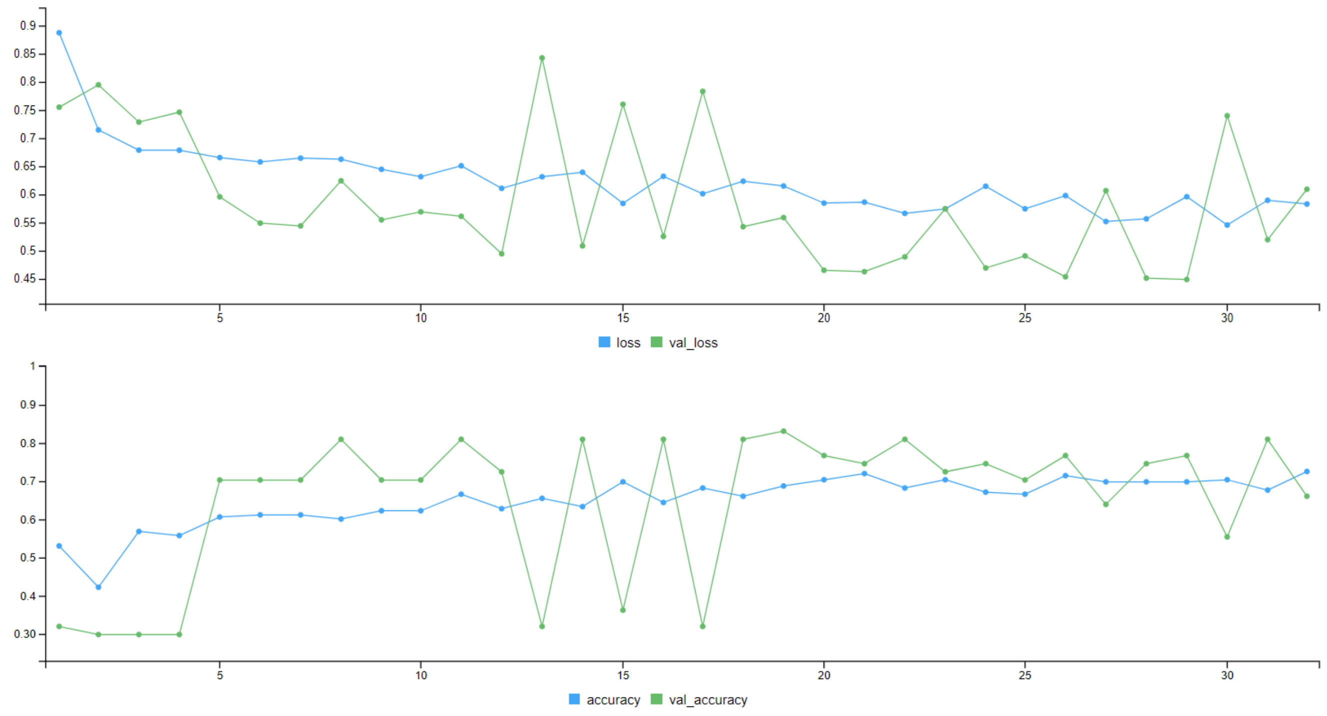
Resultados

Modelo 1 Resumen del modelo

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed)
Model: "sequential_15"
________________________________________________________________________________________________
Layer (type)                            Output Shape                          Param #
================================================================================================
dense_54 (Dense)                        (None, 16)                            78368
________________________________________________________________________________________________
dense_53 (Dense)                        (None, 16)                            272
________________________________________________________________________________________________
dense_52 (Dense)                        (None, 1)                             17
================================================================================================
Total params: 78,657
Trainable params: 78,657
Non-trainable params: 0
________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.5348775 0.7049180
```

```
> history
```

```
Final epoch (plot to see history):
        loss: 0.519
    accuracy: 0.7494
    val_loss: 0.6234
val_accuracy: 0.7216

> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed, ExpGenTCGA
           [,1]
 [1,] 0.58461225
 [2,] 0.60310704
 [3,] 0.67405194
 [4,] 0.81585377
 [5,] 0.31713414
 [6,] 0.14554837
 [7,] 0.34794801
 [8,] 0.29168567
 [9,] 0.58980548
[10,] 0.61459279
[11,] 0.48619995
[12,] 0.46376172
[13,] 0.24309367
[14,] 0.22715920
[15,] 0.49996880
[16,] 0.48184609
[17,] 0.20491400
[18,] 0.66487873
[19,] 0.28882766
[20,] 0.46932176
[21,] 0.42240679
[22,] 0.63178110
[23,] 0.34535626
[24,] 0.58418208
[25,] 0.15185118
[26,] 0.61035919
[27,] 0.53604364
[28,] 0.54063535
```

Modelo 2 Resumen del modelo

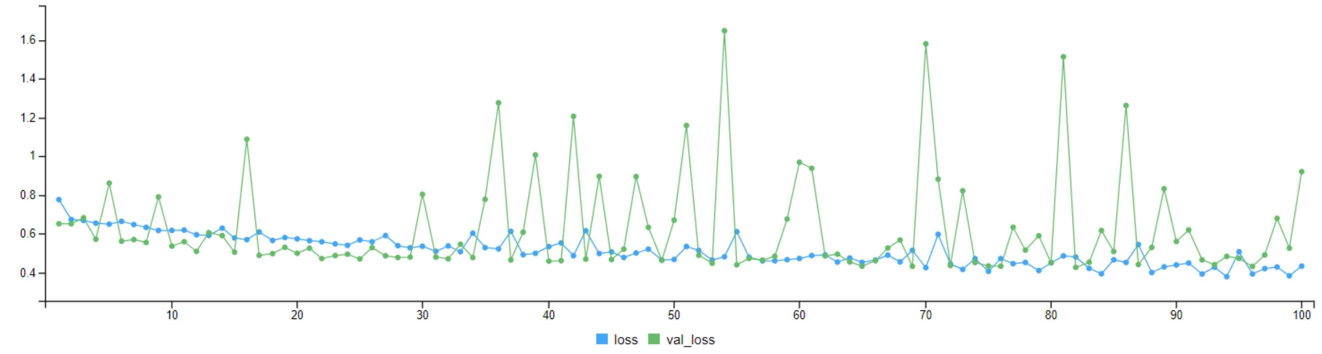
Vamos a aumentar las epochs de 32 a 100.

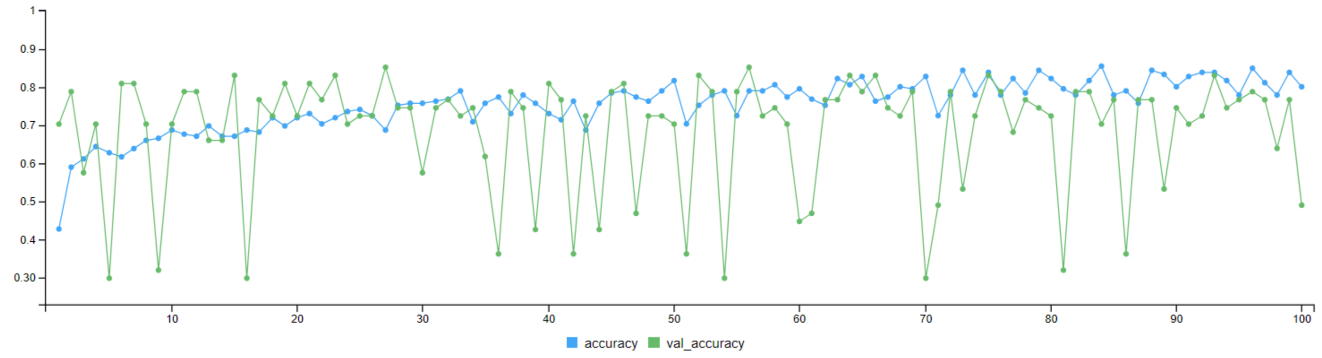
```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed)
Model: "sequential_16"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                              Param #
====================================================================================================
dense_57 (Dense)                             (None, 16)                                78368
____________________________________________________________________________________________________
dense_56 (Dense)                             (None, 16)                                272
____________________________________________________________________________________________________
dense_55 (Dense)                             (None, 1)                                 17
====================================================================================================
Total params: 78,657
```

```
Trainable params: 78,657
Non-trainable params: 0
-----------------------------------------------------------------------------------------------------
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.9658350 0.5344828
> history

Final epoch (plot to see history):
        loss: 0.4316
    accuracy: 0.8
    val_loss: 0.9197
val_accuracy: 0.4894
> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed, ExpGenTCGA
              [,1]
 [1,] 3.161165e-01
 [2,] 4.803107e-01
 [3,] 8.270217e-01
 [4,] 7.530786e-01
 [5,] 2.729392e-01
 [6,] 9.059221e-03
 [7,] 2.192812e-01
 [8,] 2.647204e-01
 [9,] 4.573123e-01
[10,] 7.129636e-01
[11,] 4.777987e-01
```

```
[12,] 5.553334e-01
[13,] 5.959237e-02
[14,] 3.188428e-02
[15,] 3.768789e-01
[16,] 2.820298e-01
[17,] 3.953472e-02
[18,] 7.984113e-01
[19,] 1.953096e-01
[20,] 1.101498e-01
[21,] 2.716530e-01
[22,] 5.827190e-01
[23,] 2.029104e-01
[24,] 3.827968e-01
[25,] 1.153731e-02
[26,] 7.944947e-01
[27,] 1.018376e-01
[28,] 9.481812e-02
```

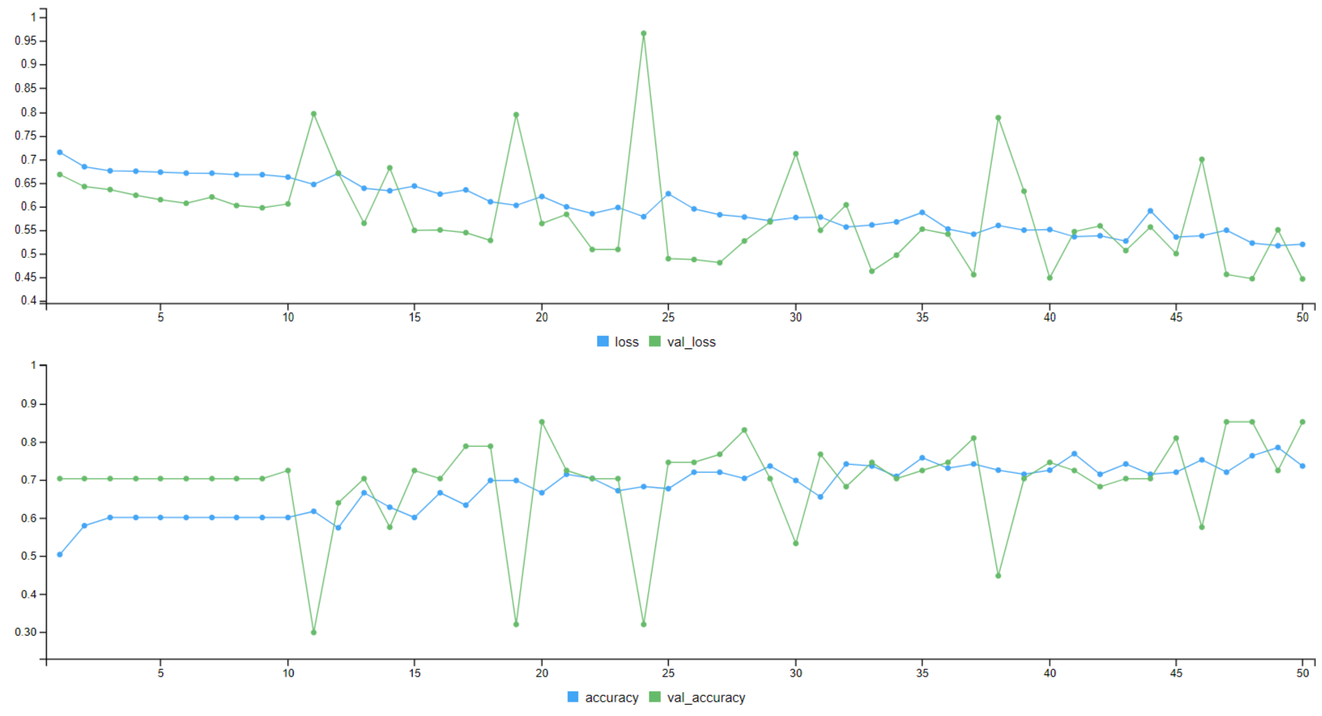
Modelo 3 Resumen del modelo

Vamos a cambiar las epochs= 50.

```
> summary(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed)
Model: "sequential_17"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                            Param #
====================================================================================================
dense_61 (Dense)                             (None, 20)                              97960
____________________________________________________________________________________________________
dense_60 (Dense)                             (None, 16)                              336
____________________________________________________________________________________________________
dense_59 (Dense)                             (None, 8)                               136
____________________________________________________________________________________________________
dense_58 (Dense)                             (None, 1)                               9
====================================================================================================
Total params: 98,441
Trainable params: 98,441
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.5299286 0.7413793
> history

Final epoch (plot to see history):
        loss: 0.5191
    accuracy: 0.7351
    val_loss: 0.4457
val_accuracy: 0.8511
> predict(Model_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_Transposed, ExpGenTCGA
            [,1]
 [1,] 0.83764637
 [2,] 0.77914727
 [3,] 0.87687790
 [4,] 0.96164382
 [5,] 0.48738194
 [6,] 0.19217414
 [7,] 0.55559862
 [8,] 0.53899276
 [9,] 0.81982958
[10,] 0.90991068
[11,] 0.68609488
[12,] 0.67460680
[13,] 0.49418601
[14,] 0.40798798
[15,] 0.75500524
[16,] 0.72544819
[17,] 0.28343984
[18,] 0.92997670
```

```
[19,] 0.49273008
[20,] 0.69549948
[21,] 0.58153230
[22,] 0.90136254
[23,] 0.54840910
[24,] 0.83853483
[25,] 0.36407518
[26,] 0.87791800
[27,] 0.82587719
[28,] 0.60410601
```

Metilación

Modelo de metilación (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt7 sondas = 93346, n = 483)

Usamos: MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed)
[1] 93346   483
```

Creación de conjuntos test y train

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed <- t(MetTCGA_KIRC_RawDa

set.seed(231)

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Index_Training <- sampl

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Test <- MetTCGA_KIRC_Ra
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Train <- MetTCGA_KIRC_I
```

Obtención de etiquetas

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Test <- MetTCGA_KIRC_RawDa
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Train <- MetTCGA_KIRC_RawDa

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Test_FactorNumb <- as.inte
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Test_FactorNumb[MetTCGA_KI

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Train_FactorNumb <- as.int
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Train_FactorNumb[MetTCGA_K
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed, file = "MetTCGA_
```

Conjuntos Train y Test

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Train, file = "Me

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Test, file = "Met
```

Etiquetas

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Train_FactorNumb, fil

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Test_FactorNumb, file
```

Creando script (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75mode

```
# Cargando los archivos

# test_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Test.rda")

# train_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Train.rda")

# test_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Test_FactorNumb.rda"

# train_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Labels_Train_FactorNumb.rda

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

# Definición del modelo MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed: sondas =
Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed <- keras_model_s
  layer_dense(units = 16, activation = "relu", input_shape = c(93346)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed %>% f
plot(history)

score <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed %>% eva
 MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed_Test, MetTCGA_KIRC_Ra
  verbose = 0
)

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

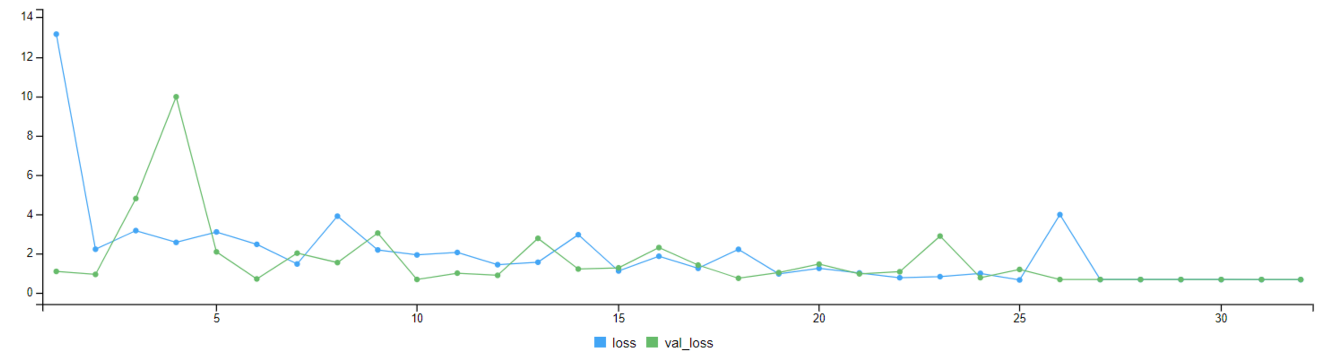
Resultados

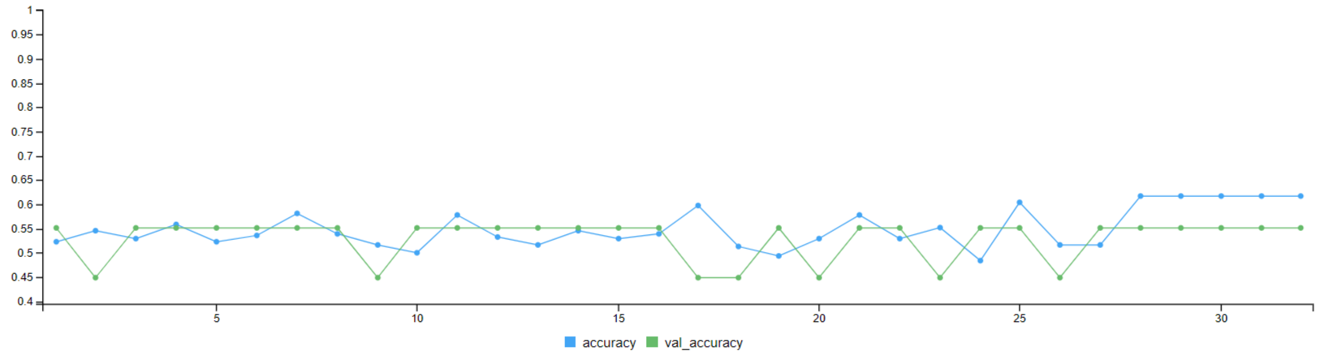
Modelo 1 Resumen del modelo

```
> summary(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed)
Model: "sequential_18"
____________________________________________________________________________________________________
Layer (type)                                  Output Shape                                Param #
====================================================================================================
dense_64 (Dense)                              (None, 16)                                  1493552
____________________________________________________________________________________________________
dense_63 (Dense)                              (None, 16)                                  272
____________________________________________________________________________________________________
dense_62 (Dense)                              (None, 1)                                   17
====================================================================================================
Total params: 1,493,841
Trainable params: 1,493,841
Non-trainable params: 0
____________________________________________________________________________________________________
>
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.6819232 0.6907216
> history

Final epoch (plot to see history):
        loss: 0.6848
    accuracy: 0.6169
    val_loss: 0.6905
val_accuracy: 0.5513
> predict(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed, MetTC
           [,1]
 [1,] 0.5153267
 [2,] 0.5153267
 [3,] 0.5153267
 [4,] 0.5153267
 [5,] 0.5153267
 [6,] 0.5153267
 [7,] 0.5153267
 [8,] 0.5153267
 [9,] 0.5153267
[10,] 0.5153267
[11,] 0.5153267
[12,] 0.5153267
[13,] 0.5153267
[14,] 0.5153267
[15,] 0.5153267
[16,] 0.5153267
[17,] 0.5153267
[18,] 0.5153267
[19,] 0.5153267
```

```
[20,] 0.5153267
[21,] 0.5153267
[22,] 0.5153267
[23,] 0.5153267
[24,] 0.5153267
[25,] 0.5153267
[26,] 0.5153267
[27,] 0.5153267
[28,] 0.5153267
```

Esta red parece ser muy pequeña, el modelo no aprende. Si bservamos los predicts son todos parecidos.

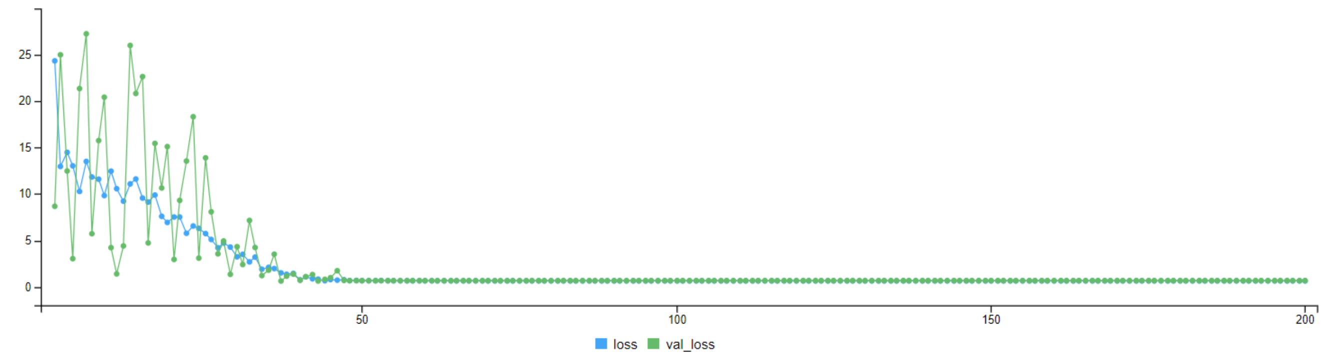
Modelo 2 Resumen del modelo

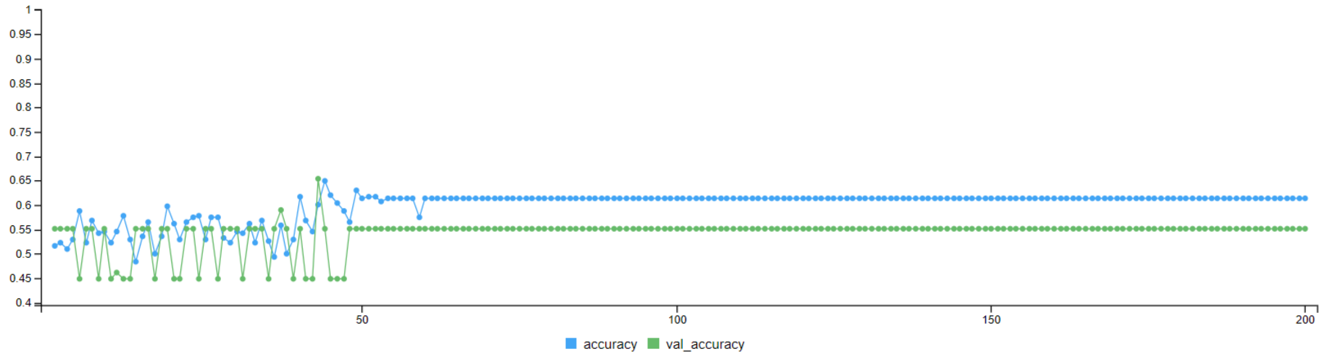
Vamos a aumentar las epochs de 32 a 200, a ver si encontramos sobreajuste.

```
> summary(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed)
Model: "sequential_20"
____________________________________________________________________________________________________
Layer (type)                                Output Shape                              Param #
====================================================================================================
dense_70 (Dense)                            (None, 16)                                1493552
____________________________________________________________________________________________________
dense_69 (Dense)                            (None, 8)                                 136
____________________________________________________________________________________________________
dense_68 (Dense)                            (None, 1)                                 9
====================================================================================================
Total params: 1,493,697
Trainable params: 1,493,697
Non-trainable params: 0
____________________________________________________________________________________________________
>
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.6313332 0.6907216
> history

Final epoch (plot to see history):
        loss: 0.6671
    accuracy: 0.6136
    val_loss: 0.696
val_accuracy: 0.5513
> predict(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed, MetTC
           [,1]
 [1,] 0.6139413
 [2,] 0.6139413
 [3,] 0.6139413
 [4,] 0.6139413
 [5,] 0.6139413
 [6,] 0.6139413
 [7,] 0.6139413
 [8,] 0.6139413
 [9,] 0.6139413
[10,] 0.6139413
[11,] 0.6139413
[12,] 0.6139413
[13,] 0.6139413
[14,] 0.6139413
[15,] 0.6139413
[16,] 0.6139413
[17,] 0.6139413
[18,] 0.6139413
```

```
[19,] 0.6139413
[20,] 0.6139413
[21,] 0.6139413
[22,] 0.6139413
[23,] 0.6139413
```

Sigue sin aprender, vamos a hacer un modelo más grande.

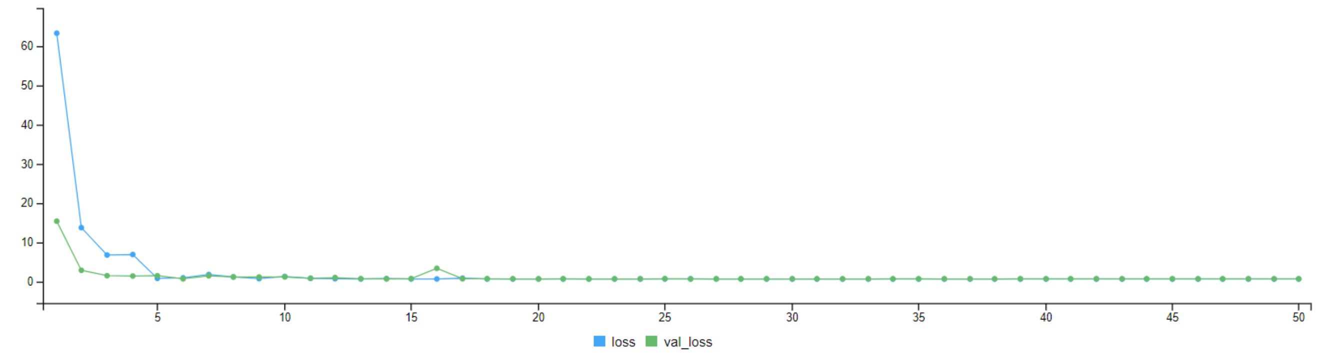
Modelo 3 Resumen del modelo

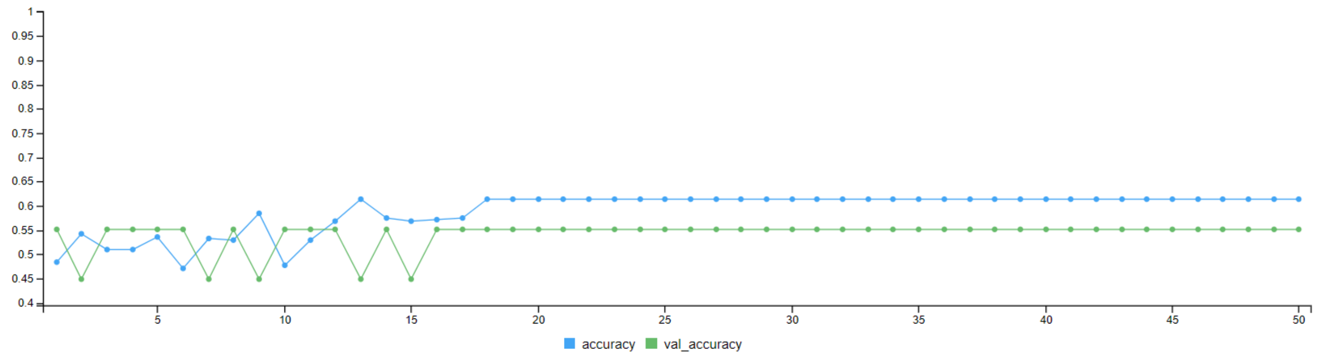
Vamos a aumentar las epochs de 32 a 200, a ver si encontramos sobreajuste.

```
> summary(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed)
Model: "sequential_22"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                                Param #
====================================================================================================
dense_81 (Dense)                             (None, 500)                                 46673500
____________________________________________________________________________________________________
dense_80 (Dense)                             (None, 200)                                 100200
____________________________________________________________________________________________________
dense_79 (Dense)                             (None, 100)                                 20100
____________________________________________________________________________________________________
dense_78 (Dense)                             (None, 50)                                  5050
____________________________________________________________________________________________________
dense_77 (Dense)                             (None, 20)                                  1020
____________________________________________________________________________________________________
dense_76 (Dense)                             (None, 1)                                   21
====================================================================================================
Total params: 46,799,891
Trainable params: 46,799,891
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.6286385 0.6907216
> history

Final epoch (plot to see history):
        loss: 0.6682
    accuracy: 0.6136
    val_loss: 0.6985
val_accuracy: 0.5513
> predict(Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_Transposed, MetTC
           [,1]
 [1,] 0.6227190
 [2,] 0.6227190
 [3,] 0.6227190
 [4,] 0.6227190
 [5,] 0.6227190
 [6,] 0.6227190
 [7,] 0.6227190
 [8,] 0.6227190
 [9,] 0.6227190
[10,] 0.6227190
[11,] 0.6227190
[12,] 0.6227190
[13,] 0.6227190
[14,] 0.6227190
[15,] 0.6227190
[16,] 0.6227190
[17,] 0.6227190
[18,] 0.6227190
[19,] 0.6227190
```

```
[20,] 0.6227190
[21,] 0.6227190
[22,] 0.6227190
[23,] 0.6227190
[24,] 0.6227190
[25,] 0.6227190
[26,] 0.6227190
[27,] 0.6227190
[28,] 0.6227190
```

Sigue sin aprender.

Modelo de metilación (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt7
sondas = 93346, n = 291)

Usamos: MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75___SameSampExpProt_Ren

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed)
[1] 93346   483
```

Creación de conjuntos test y train

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed <- t(

set.seed(231)

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_Index

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_Test
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_Train
```

Obtención de etiquetas

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Test <- M
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Train <-

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Test_Fact
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Test_Fact

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Train_Fac
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Train_Fac
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed,
```

Conjuntos Train y Test

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_T

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_T
```

Etiquetas

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Trai

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Test_
```

Creando script (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Sam

```
# Cargando los archivos

# test_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_

# train_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_

# test_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Tes

# train_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Labels_Tra

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

# Definición del modelo MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_
Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed
  layer_dense(units = 16, activation = "relu", input_shape = c(93346)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Trans
plot(history)

score <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Tr
 MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_Transposed_Test
  verbose = 0
)

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

Modelo de metilación (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt7
sondas = 93346, n = 290)

Usamos: MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameS

```
> dim(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
[1] 93346   290
```

Creación de conjuntos test y train

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Tra

set.seed(231)

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Ind

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Tra
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Tra
```

Obtención de etiquetas

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab

MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_Lab
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Rename
```

Conjuntos Train y Test

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
```

Etiquetas

```
# Train
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

# Test
save(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
```

Creando script (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_San

```
# Cargando los archivos

# test_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

# train_x
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

# test_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

# train_y
load("MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam

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

# Definición del modelo MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_S
Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
  layer_dense(units = 16, activation = "relu", input_shape = c(93346)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renam
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSamp
plot(history)

score <- Model_MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExp
 MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_T
  verbose = 0
)

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

Proteómica

Modelo de Expresión proteica (ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed: proteínas = 177, n = 478)

Usamos: ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed

```
> dim(ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed)
[1] 177 478
```

Dado que de este conjunto de datos no podemos obtener todas las etiquetas de vital_status para las 478 muestras, porque no se pueden descargar este tipo de metadatos, pasaremos al siguiente modelo en el que tenemos las muestras que coinciden entre expresión génica y expresión proteica. Así, podremos conocer las etiquetas de las muestras gracias a los datos de Expresión génica.

Modelo de Expresión proteica (ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Rena proteínas = 177, n = 474)

Usamos: ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed

```
 ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed <- ExpProtTCGA_KIRC_RawData_woNA_Norm_Rename
> dim(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed)
[1] 177 474
```

Creación de conjuntos test y train

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed <- t(ExpProtTCGA_KIRC_RawData_woNA

set.seed(231)

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Index_Training<- sample(1:nrow(Exp

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Test <- ExpProtTCGA_KIRC_RawData_w
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Train <- ExpProtTCGA_KIRC_RawData_
```

Obtención de etiquetas

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels <- ExpProtTCGA_KIRC_RawData_woNA_Norm_
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test <- ExpProtTCGA_KIRC_RawData_woNA_
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train <- ExpProtTCGA_KIRC_RawData_woNA_

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test_FactorNumb <- as.integer(factor(E
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test_FactorNumb[ExpProtTCGA_KIRC_RawDa

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train_FactorNumb <- as.integer(factor(
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train_FactorNumb[ExpProtTCGA_KIRC_RawD
```

Guardar objetos importantes para modelos

Base de datos completa

```
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed, file = "ExpProtTCGA_KIRC_Raw
```

Conjuntos Train y Test

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Train, file = "ExpProtTCGA_KI

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Test, file = "ExpProtTCGA_KIR
```

Etiquetas

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train_FactorNumb, file = "ExpProt

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test_FactorNumb, file = "ExpProtT
```

Creando script (ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGenmmodelscript.R)

```
# Cargando los archivos

# test_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Test.rda")

# train_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Train.rda")

# test_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test_FactorNumb.rda")

# train_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train_FactorNumb.rda")
```

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

# Definición del modelo ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed: proteínas = 177, n =
Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed <- keras_model_sequential() %
  layer_dense(units = 16, activation = "relu", input_shape = c(177)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

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

# Entrenamiento y evaluación del modelo

history <- Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed %>% fit(ExpProtTC

plot(history)

score <-  Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed %>% evaluate(
 ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Test, ExpProtTCGA_KIRC_RawData_wo
  verbose = 0
)

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

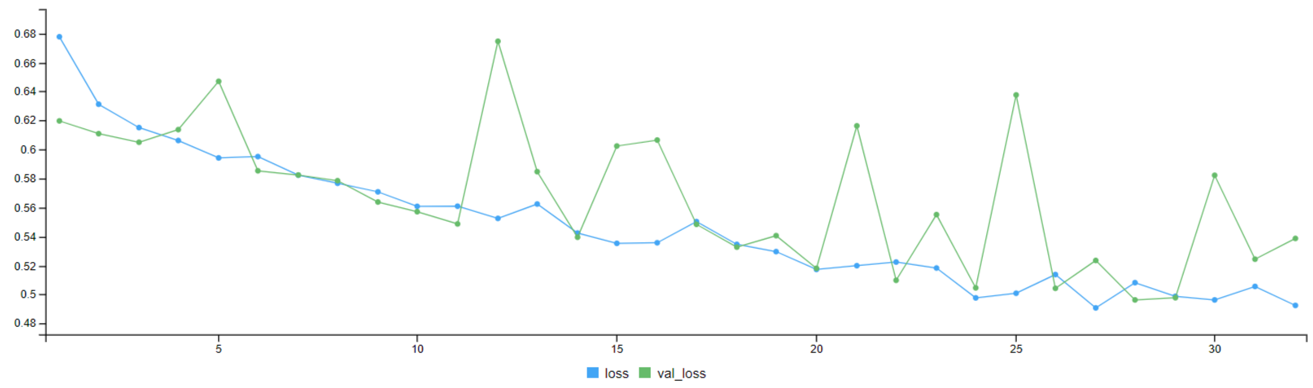
Resultados

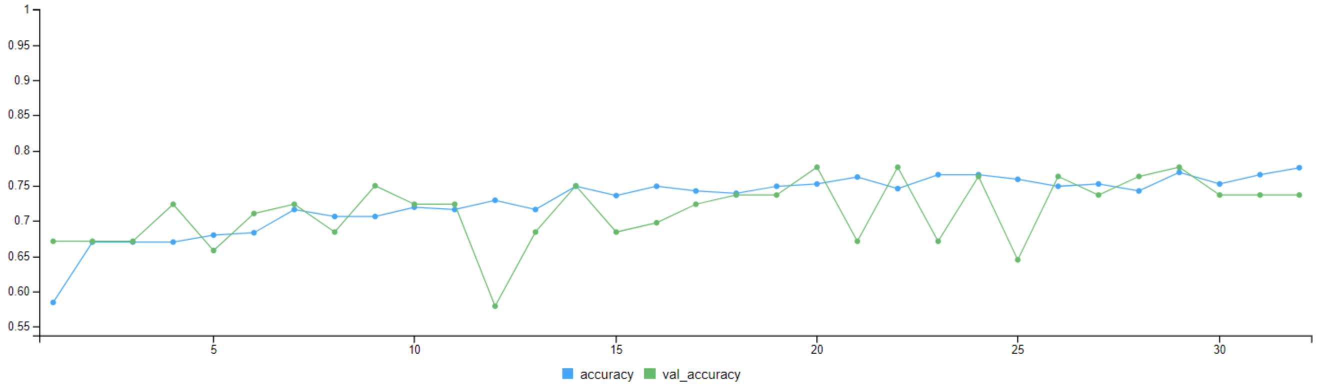
Modelo 1 Resumen del modelo

```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed)
Model: "sequential_3"
____________________________________________________________________________________________________
Layer (type)                                Output Shape                                Param #
====================================================================================================
dense_11 (Dense)                            (None, 16)                                  2848
____________________________________________________________________________________________________
dense_10 (Dense)                            (None, 16)                                  272
____________________________________________________________________________________________________
dense_9 (Dense)                             (None, 1)                                   17
====================================================================================================
Total params: 3,137
Trainable params: 3,137
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.7878002 0.6000000
> history

Final epoch (plot to see history):
        loss: 0.4925
    accuracy: 0.7756
    val_loss: 0.5387
val_accuracy: 0.7368
> predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed, ExpProtTCGA_KIRC_
           [,1]
 [1,] 0.8020363
 [2,] 0.7002377
 [3,] 0.8389700
 [4,] 0.9240683
 [5,] 0.6572965
 [6,] 0.7046134
 [7,] 0.5297390
 [8,] 0.9204122
 [9,] 0.5820349
[10,] 0.9205120
[11,] 0.9140373
[12,] 0.3199417
[13,] 0.9347509
[14,] 0.9449486
[15,] 0.6863053
```

```
[16,] 0.7290468
[17,] 0.6065519
[18,] 0.8446778
[19,] 0.7743371
[20,] 0.9493220
[21,] 0.9058605
[22,] 0.7501490
[23,] 0.6700755
[24,] 0.6719083
[25,] 0.8489730
[26,] 0.8599136
[27,] 0.9141070
[28,] 0.7553965
```

Parece que hay overfitting, vamos a poner dropout.

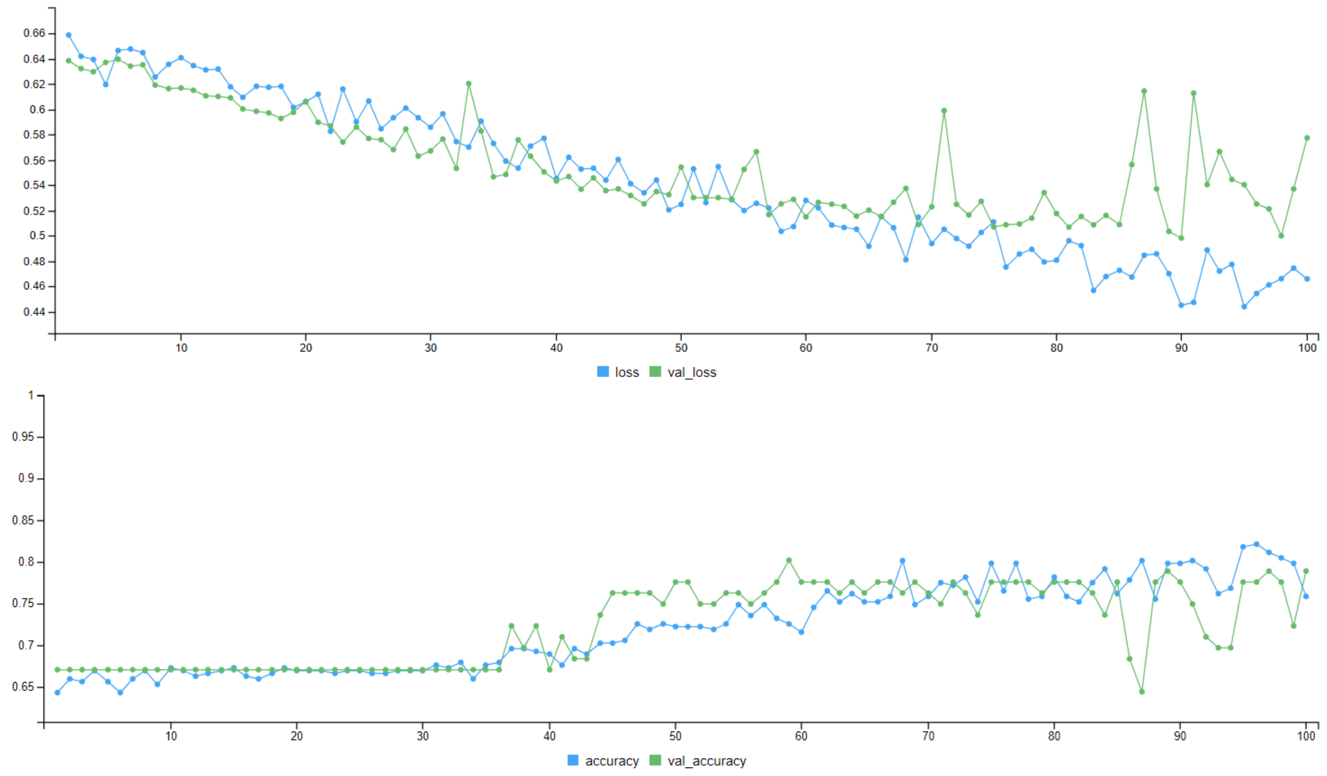
Modelo 2 Resumen del modelo

Aumentamos las capas y añadimos dropout.

```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed)
Model: "sequential"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                            Param #
====================================================================================================
dense_3 (Dense)                              (None, 20)                              3560
____________________________________________________________________________________________________
dropout_2 (Dropout)                          (None, 20)                              0
____________________________________________________________________________________________________
dense_2 (Dense)                              (None, 16)                              336
____________________________________________________________________________________________________
dropout_1 (Dropout)                          (None, 16)                              0
____________________________________________________________________________________________________
dense_1 (Dense)                              (None, 8)                               136
____________________________________________________________________________________________________
dropout (Dropout)                            (None, 8)                               0
____________________________________________________________________________________________________
dense (Dense)                                (None, 1)                               9
====================================================================================================
Total params: 4,041
Trainable params: 4,041
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:



```
> score
     loss  accuracy
0.8679943 0.6000000
> history

Final epoch (plot to see history):
        loss: 0.4659
    accuracy: 0.7591
    val_loss: 0.5773
val_accuracy: 0.7895
> predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed, ExpProtTCGA_KIRC_
           [,1]
 [1,] 0.7436856
 [2,] 0.8600804
 [3,] 0.8964235
 [4,] 0.9579632
 [5,] 0.5083777
 [6,] 0.5475296
 [7,] 0.3687240
 [8,] 0.9250146
 [9,] 0.4088703
[10,] 0.9768463
[11,] 0.9818953
[12,] 0.4274961
[13,] 0.9851228
[14,] 0.9495082
[15,] 0.9644765
[16,] 0.6614694
```

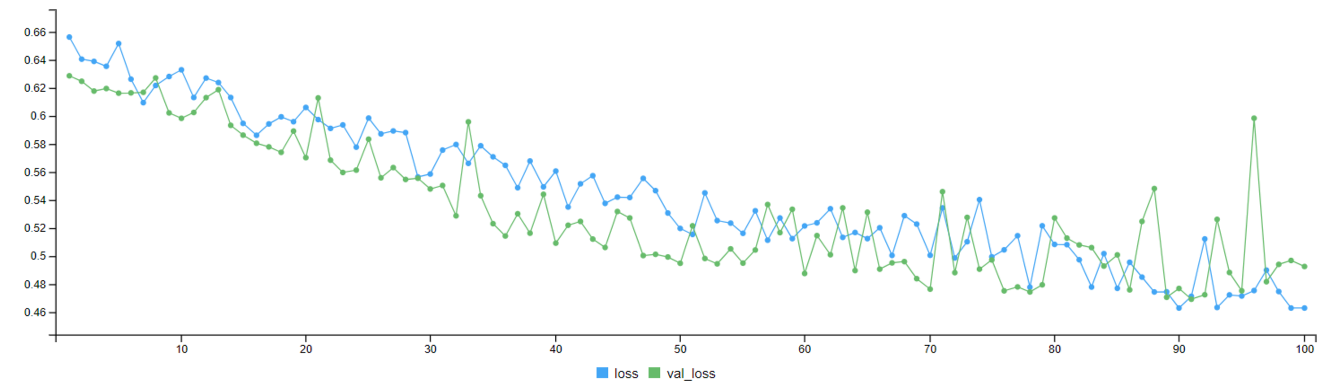
```
[17,] 0.7163674
[18,] 0.8811067
[19,] 0.7547270
[20,] 0.9704388
[21,] 0.9655876
[22,] 0.7541791
[23,] 0.5552937
[24,] 0.8054403
[25,] 0.9021704
[26,] 0.7542448
[27,] 0.9641717
[28,] 0.5106784
```

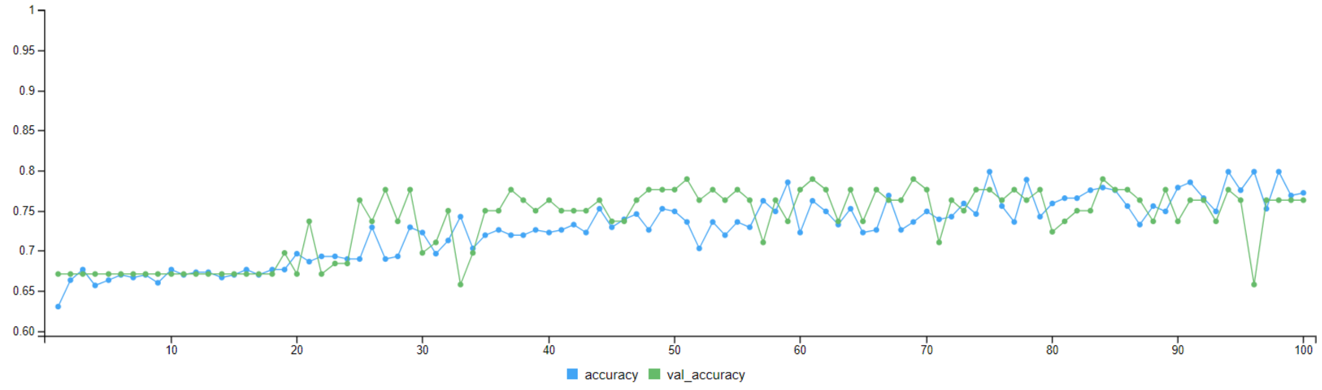
Modelo 3 Resumen del modelo

```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed)
Model: "sequential_1"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                              Param #
====================================================================================================
dense_7 (Dense)                              (None, 20)                                3560
____________________________________________________________________________________________________
dropout_5 (Dropout)                          (None, 20)                                0
____________________________________________________________________________________________________
dense_6 (Dense)                              (None, 16)                                336
____________________________________________________________________________________________________
dropout_4 (Dropout)                          (None, 16)                                0
____________________________________________________________________________________________________
dense_5 (Dense)                              (None, 8)                                 136
____________________________________________________________________________________________________
dropout_3 (Dropout)                          (None, 8)                                 0
____________________________________________________________________________________________________
dense_4 (Dense)                              (None, 1)                                 9
====================================================================================================
Total params: 4,041
Trainable params: 4,041
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.6850594 0.6631579
> history

Final epoch (plot to see history):
        loss: 0.4629
    accuracy: 0.7723
    val_loss: 0.4926
val_accuracy: 0.7632
> predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed, ExpProtTCGA_KIRC_
           [,1]
 [1,] 0.7606218
 [2,] 0.5897886
 [3,] 0.7539814
 [4,] 0.9038323
 [5,] 0.5623838
 [6,] 0.5429451
 [7,] 0.2953670
 [8,] 0.8428588
 [9,] 0.3376764
[10,] 0.8447326
[11,] 0.9222915
[12,] 0.2503030
[13,] 0.9530908
[14,] 0.9092823
[15,] 0.7956692
[16,] 0.4327042
[17,] 0.6261219
```

```
[18,] 0.8463207
[19,] 0.4852300
[20,] 0.9246504
[21,] 0.8973757
[22,] 0.6761926
[23,] 0.6136307
[24,] 0.4704547
[25,] 0.7839123
[26,] 0.8087447
[27,] 0.9226632
[28,] 0.5500689
```

**Modelo de Expresión proteica (ExpProtTCGA_KIRC_RawData_woNA_Norm___SameSampExpGen_San
proteínas = 177, n = 290)**

Usamos: ExpProtTCGA_KIRC_RawData_woNA_Norm_Renamed

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed <- ExpProtTCGA_KIRC_RawData_woNA_
> dim(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed)
[1] 177 290
```

Creación de conjuntos test y train

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed <- t(ExpProtTCGA_KIRC_

set.seed(231)

ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Index_Training<- sampl

ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Test <- ExpProtTCGA_KI
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Train <- ExpProtTCGA_K
```

Obtención de etiquetas

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test <- ExpProtTCGA_KIRC_I
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Train <- ExpProtTCGA_KIRC_

ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb <- as.inte
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb[ExpProtTCG

ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb <- as.int
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb[ExpProtTC
```

Guardar objetos importantes para modelos

Base de datos completa

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

Conjuntos Train y Test

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Train, file = "E

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Test, file = "Ex
```

Etiquetas

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb, fi

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb, fil
```

Creando script (ExpProtTCGA_KIRC_RawData_woNA_Norm___SameSampExpGen_SameSampMetscri

```
# Cargando los archivos

# test_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Test.rda")

# train_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Train.rda")

# test_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test_FactorNumb.rda

# train_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Train_FactorNumb.rd

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

# Definición del modelo ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed: proteín
Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed <- keras_model_
  layer_dense(units = 16, activation = "relu", input_shape = c(177)) %>%
  layer_dense(units = 16, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")

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

# Entrenamiento y evaluación del modelo

history <- Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed %>%
plot(history)

score <- Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed %>% eva
 ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed_Test, ExpProtTCGA_KI
  verbose = 0
)

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

Resultados

Modelo 1 Tenemos que superar el:

```
> score
     loss  accuracy
0.5824013 0.7758621
```

del modelo

```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_SameSampExpGen_SameSampMet_Renamed)
Model: "sequential_4"
________________________________________________________________________________________________
Layer (type)                              Output Shape                              Param #
================================================================================================
dense_14 (Dense)                          (None, 8)                                 1424
________________________________________________________________________________________________
dropout_1 (Dropout)                       (None, 8)                                 0
________________________________________________________________________________________________
dense_13 (Dense)                          (None, 2)                                 18
________________________________________________________________________________________________
dropout (Dropout)                         (None, 2)                                 0
________________________________________________________________________________________________
dense_12 (Dense)                          (None, 1)                                 3
================================================================================================
Total params: 1,445
Trainable params: 1,445
Non-trainable params: 0
________________________________________________________________________________________________
```

que utilizaba los datos sin normalizar.

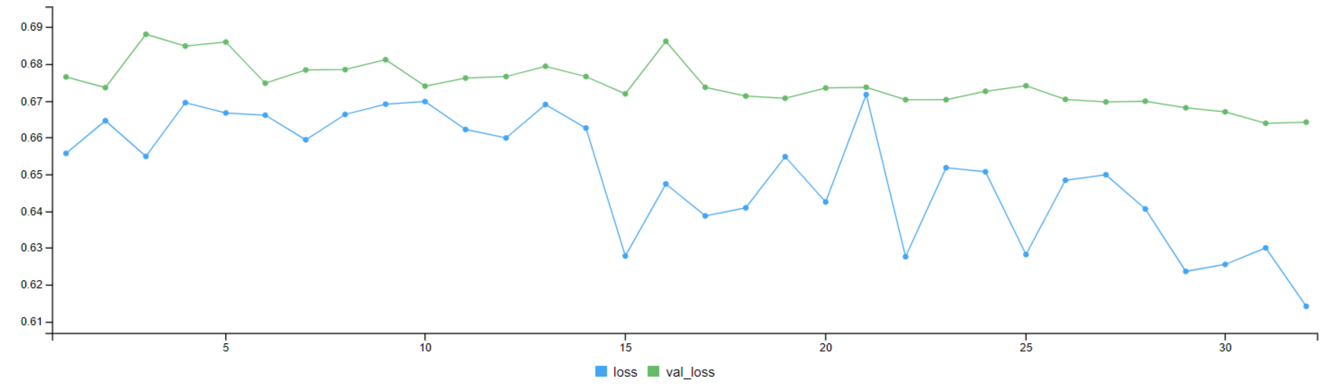
Resumen del modelo

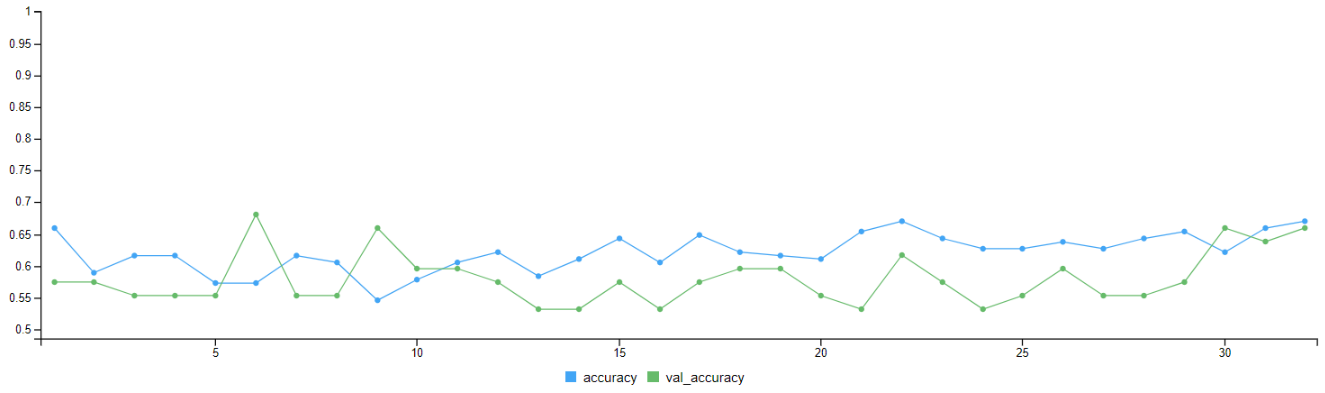
```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed)
Model: "sequential_8"
________________________________________________________________________________________________
Layer (type)                              Output Shape                              Param #
================================================================================================
dense_28 (Dense)                          (None, 8)                                 1424
________________________________________________________________________________________________
```

```
dropout_19 (Dropout)                  (None, 8)                          0
____________________________________________________________________________________________________
dense_27 (Dense)                      (None, 2)                          18
____________________________________________________________________________________________________
dropout_18 (Dropout)                  (None, 2)                          0
____________________________________________________________________________________________________
dense_26 (Dense)                      (None, 1)                          3
====================================================================================================
Total params: 1,445
Trainable params: 1,445
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.5836589 0.7413793
> history

Final epoch (plot to see history):
        loss: 0.6141
    accuracy: 0.6703
    val_loss: 0.6642
val_accuracy: 0.6596
> predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed, ExpP
           [,1]
```

```
 [1,] 0.6588583
 [2,] 0.6685524
 [3,] 0.6444975
 [4,] 0.5781231
 [5,] 0.4950803
 [6,] 0.6356037
 [7,] 0.6146865
 [8,] 0.6851306
 [9,] 0.6228296
[10,] 0.5840369
[11,] 0.6150343
[12,] 0.6080506
[13,] 0.6911544
[14,] 0.5064683
[15,] 0.6277201
[16,] 0.5299095
[17,] 0.6735159
[18,] 0.5453626
[19,] 0.5641113
[20,] 0.5420477
[21,] 0.6450393
[22,] 0.6378152
[23,] 0.5608349
[24,] 0.5863496
[25,] 0.5772927
[26,] 0.6212904
[27,] 0.6647993
[28,] 0.6734720
[29,] 0.5984204
[30,] 0.6461792
[31,] 0.5455145
[32,] 0.6058615
[33,] 0.6539574
[34,] 0.5648665
[35,] 0.6712549
[36,] 0.5809708
[37,] 0.6397321
[38,] 0.5093380
[39,] 0.5545826
[40,] 0.5814481
[41,] 0.5352771
[42,] 0.6896653
[43,] 0.5567027
[44,] 0.6017753
[45,] 0.5956193
[46,] 0.6171705
[47,] 0.6123638
[48,] 0.6179831
[49,] 0.6002995
[50,] 0.6846988
[51,] 0.5857004
[52,] 0.4995881
[53,] 0.6531309
[54,] 0.6165737
```

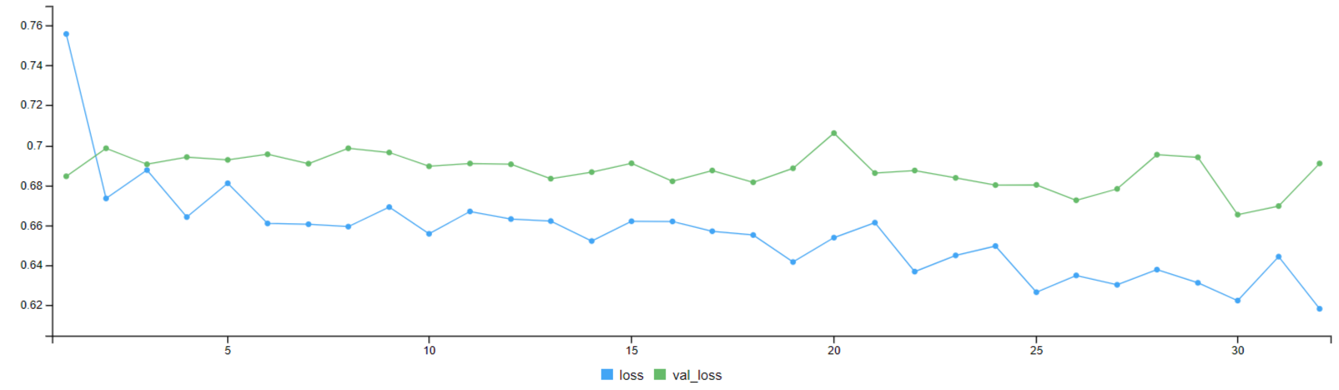
```
[55,] 0.6023639
[56,] 0.6046523
[57,] 0.5799499
[58,] 0.6944157
```

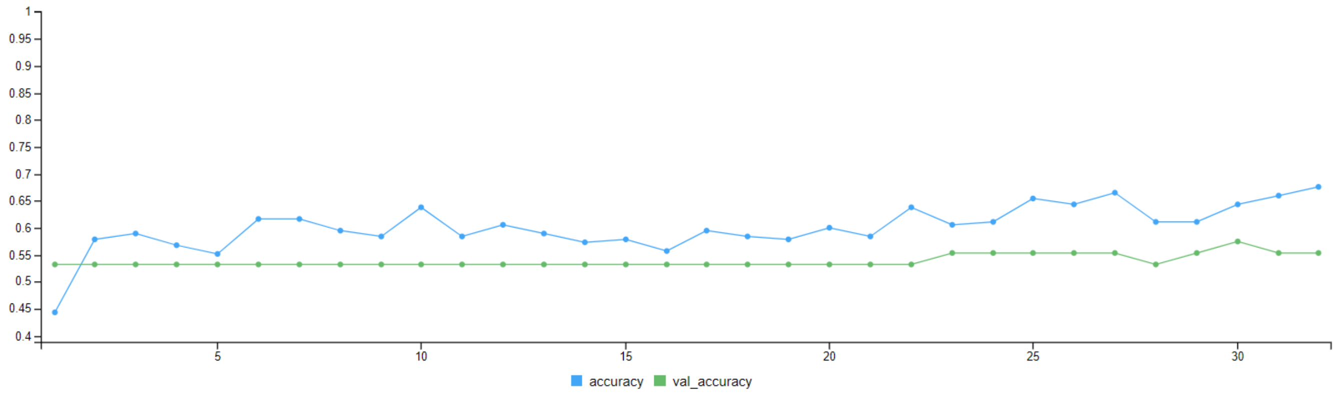
Modelo 2 Resumen del modelo

```
> summary(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed)
Model: "sequential_7"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                             Param #
====================================================================================================
dense_25 (Dense)                             (None, 10)                               1780
____________________________________________________________________________________________________
dropout_17 (Dropout)                         (None, 10)                               0
____________________________________________________________________________________________________
dense_24 (Dense)                             (None, 4)                                44
____________________________________________________________________________________________________
dropout_16 (Dropout)                         (None, 4)                                0
____________________________________________________________________________________________________
dense_23 (Dense)                             (None, 1)                                5
====================================================================================================
Total params: 1,829
Trainable params: 1,829
Non-trainable params: 0
____________________________________________________________________________________________________
```

Gráficas de pérdida y precisión

Ahora, presentamos las gráficas de pérdida y precisión:





```
> score
     loss  accuracy
0.6397267 0.7758621
> history

Final epoch (plot to see history):
        loss: 0.6203
    accuracy: 0.7135
    val_loss: 0.6594
val_accuracy: 0.6383
> predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed, ExpP
           [,1]
 [1,] 0.5676859
 [2,] 0.5676859
 [3,] 0.5676859
 [4,] 0.5676859
 [5,] 0.5083192
 [6,] 0.5676859
 [7,] 0.5676859
 [8,] 0.5676859
 [9,] 0.5676859
[10,] 0.5662532
[11,] 0.5676859
[12,] 0.5676859
[13,] 0.5529552
[14,] 0.5415108
[15,] 0.5676859
[16,] 0.5676859
```

```
[17,] 0.5676859
[18,] 0.5676859
[19,] 0.2972086
[20,] 0.5367039
[21,] 0.5676859
[22,] 0.5676859
[23,] 0.5676859
[24,] 0.5439425
[25,] 0.5676859
[26,] 0.5676859
[27,] 0.5676859
[28,] 0.5676859
[29,] 0.4727215
[30,] 0.5676859
[31,] 0.3027540
[32,] 0.5676859
[33,] 0.5676859
[34,] 0.4062908
[35,] 0.5676859
[36,] 0.5676859
[37,] 0.5676859
[38,] 0.3041144
[39,] 0.5676859
[40,] 0.5565755
[41,] 0.4916607
[42,] 0.5676859
[43,] 0.4987919
[44,] 0.5589045
[45,] 0.5676859
[46,] 0.5661155
[47,] 0.5676859
[48,] 0.5676859
[49,] 0.5676859
[50,] 0.5676859
[51,] 0.5479358
[52,] 0.3945759
[53,] 0.5676859
[54,] 0.5676859
[55,] 0.5615835
[56,] 0.5355625
[57,] 0.5676859
[58,] 0.5676859

> which(predict(Model_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Transposed
[1] 19 29 31 34 38 41 43 52
> ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_Labels_Test[c(19,29,31,34,38,4
[1] "Dead"  "Dead"  "Alive" "Dead"  "Alive" "Dead"  "Alive" "Alive"
```

De las 8 muestras que el predict del modelo dice que son de pacientes muertos, solo en 4 está en lo cierto.

7. Support Vector Machine (SVM)

Para saber si estos modelos pueden mejorar su precisión podemos realizar un SVM con kernel radial basis function (rbf). Para esto necesitaremos modificar un poco los datasets de train y test que ya habíamos creado en el apartado **6. Modelo independientes de ómicas** añadiéndoles una columna llamada labels en una columna. Tendremos que instalar el paquete `e1071` y utilizar la función `svm()`.

```
install.packages("e1071")
library(e1071)
```

Expresión Génica

Modelo de Expresión génica (ExpGenTCGA__KIRC__Norm01__Filt75__Renamed: genes = 4897, n = 606)

Preparación de dataset El conjunto train de este modelo: ExpGenTCGA__KIRC__Norm01__Filt75__Renamed__Transpose
Precisión final del SVM: 71.31%

```
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM <- cbind(ExpGenTCGA_KIRC_Norm01_Filt75_Renam
colnames(ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM_classifierR = svm(formula = Labels ~ .,
                 data = ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

> ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM_classifierR

Call:
svm(formula = Labels ~ ., data = ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM,
    type = "C-classification", kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  386
```

Predicciones

```
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM_classifierR_predR = predict(ExpGenTCGA_KIRC_
```

Matriz de confusión

```
cmR_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM_classifierR_predR = table(ExpGenTCGA_KIRC
> cmR_ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Transposed_Train_SVM_classifierR_predR
                                                            ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Trans
ExpGenTCGA_KIRC_Norm01_Filt75_Renamed_Labels_Test_FactorNumb  0  1
                                                           0  8 32
                                                           1  3 79
> (79+8)/(8+32+3+79)
[1] 0.7131148
```

Modelo de Expresión génica (ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt__Renamed: genes = 4897, n = 474)

Preparación de dataset El conjunto train de este modelo: ExpGenTCGA__KIRC__Norm01__Filt75__Renamed__Transpose
Precisión final del SVM: 66.31%

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM <- cbind(ExpGenTCGA_KIRC_Norm01_Filt75_SameSa
colnames(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR = svm(formula = Labels ~ .,
                 data = ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

> ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR

Call:
svm(formula = Labels ~ ., data = ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM,
    type = "C-classification", kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  299
```

Predicciones

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR_predR = predict(ExpGenTCGA_KIRC_N
```

Matriz de confusión

```
cmR_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR_predR = table(ExpGenTCGA_KIRC
> cmR_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR_predR
                                                                           ExpGenTCGA_KIRC_Norm01_Filt7
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_SVM_classifierR_predR  0  1
                                                                          0  6  5
                                                                          1 27 57
> (57+6)/(57+6+5+27)
[1] 0.6631579
```

**Modelo de Expresión génica (ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt__SameSampMet_
genes = 4897, n = 290)**

Preparación de dataset El conjunto train de este modelo: ExpGenTCGA__KIRC__Norm01__Filt75__SameSampExpProt_
Precisión final del SVM: 77.58%

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM <- cbind(ExpGenTCGA_KIRC_Norm01_Fi
colnames(ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR = svm(formula = Labels
                 data = ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

> ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR

Call:
svm(formula = Labels ~ ., data = ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM,
    type = "C-classification", kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  198
```

Predicciones

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR_predR = predict(ExpGen
```

Matriz de confusión

```
cmR_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR_predR = table(Exp
> cmR_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR_predR
                                                                                      ExpGenTCGA_KIRC_
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_SameSampMet_Renamed_SVM_classifierR_predR  0  1
                                                                                     0  7  6
                                                                                     1  7 38
> (38+7)/(38+7+6+7)
[1] 0.7758621
```

Metilación

**Modelo de metilación (MetTCGA__KIRC__RawData__woDupSamples__woSNP__woXY__woNA__Norm__Filt7
sondas = 93346, n = 483)**

Preparación de dataset El conjunto train de este modelo: MetTCGA__KIRC__RawData__woDupSamples__woSNP__woXY
Precisión final del SVM: No puedo crear el SVM porque no hay RAM suficiente

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_SVM <- cbind(MetTCGA_KIRC_RawData_
colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_SVM_classifierR = svm(formula = La
                 data = MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_Renamed_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

**Error: cannot allocate vector of size 32.5 Gb**
```

Modelo de metilación (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt7 sondas = 93346, n = 291)

Preparación de dataset El conjunto train de este modelo: MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY
Precisión final del SVM: No puedo crear el SVM porque no hay RAM suficiente

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_SVM <- cbind(Met
colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_SVM)[1]
```

Entrenamiento del modelo

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_Renamed_SVM_classifierR 
                 data = MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75__SameSampExpProt_
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')
Error: cannot allocate vector of size 32.5 Gb
```

Modelo de metilación (MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt7 sondas = 93346, n = 290)

Preparación de dataset El conjunto train de este modelo: MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY
Precisión final del SVM: No puedo crear el SVM porque no hay RAM suficiente

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_SV
colnames(MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_R
```

Entrenamiento del modelo

```
MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_SameSampExpGen_Renamed_SV
                 data = MetTCGA_KIRC_RawData_woDupSamples_woSNP_woXY_woNA_Norm_Filt75_SameSampExpProt_S
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')
Error: cannot allocate vector of size 32.5 Gb
```

Proteómica

Modelo de Expresión proteica (ExpProtTCGA__KIRC__RawData__woNA__Norm__SameSampExpGen__Rena
proteínas = 177, n = 474)

Preparación de dataset El conjunto train de este modelo: ExpProtTCGA_KIRC_RawData_woNA_Norm___SameSamp
Precisión final del SVM: 77.58%

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM <- cbind(ExpProtTCGA_KIRC_Raw
colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR = svm(formula = 
                 data = ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

> ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR

Call:
svm(formula = Labels ~ ., data = ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_
    type = "C-classification", kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  212
```

Predicciones

```
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR_predR = predict(
```

Matriz de confusión

```
cmR_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR_predR = table
> cmR_ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR_predR
                                                                                           ExpProtTCGA_
ExpProtTCGA_KIRC_RawData_woNA_Norm__SameSampExpGen_SameSampMet_Renamed_SVM_classifierR_predR  0  1
                                                                                          0  5  5
                                                                                          1  8 40
> (40+5)/(40+5+8+5)
[1] 0.7758621
```

Modelo de Expresión proteica (ExpProtTCGA__KIRC__RawData__woNA__Norm___SameSampExpGen__San
proteínas = 177, n = 290)

Preparación de dataset El conjunto train de este modelo: ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampE
Precisión final del SVM: 63.16%

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM <- cbind(ExpProtTCGA_KIRC_RawData_woNA_No
colnames(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR = svm(formula = Labels ~ .,
                 data = ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')

> ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR

Call:
svm(formula = Labels ~ ., data = ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM,
    type = "C-classification", kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  307
```

Predicciones

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR_predR = predict(ExpProtTCGA_K
```

Matriz de confusión

```
cmR_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR_predR = table(ExpProtTCGA
> cmR_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR_predR
                                                                                  ExpProtTCGA_KIRC_RawData
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_SVM_classifierR_predR  0  1
                                                                               0  9  4
                                                                               1 31 51
> (51+9)/(51+9+4+31)
[1] 0.6315789
```

8. Modelos de ómicas integradas

Modelo de dos ómicas

En este caso haremos dos modelos con los datos de proteómica y transcriptómica integrados. Los dos modelos se basan en que antes de pasar los datos por el clasificador serán el input de un autoencoder que reducirá la dimensionalidad (columnas/genes/proteínas) de los datasets. Los autoencoders están formados por un codificador, que comprime los datos y un decodificador que intenta descomprimir los datos lo mejor posible para que los datos de entrada y salida sean iguales. La mayor utilidad de los autoencoders se ha encontrado en la reducción de la dimensionalidad o en modelos de deep learning generativos. Tras comprobar que los datos de salida del autoencoder son similares o iguales a los datos de entrada podremos “desensamblar” el autoencoder para quedarnos solo con el encoder, más concretamente con los el output del bottleneck del autoencoder.

Este bottleneck que tendrá la dimensionalidad reducida pero con la información más relevante de nuestro dataset de ómicas integradas se utilizará como entrada para nuestro clasificador.

En este caso queremos hacer dos tipos de autoencoder:

1) Autoencoder que tiene como input un dataset con la unión de los datos de proteómica y transcriptómica mediante un cbind()
2) Autoencoder que tiene como input los datasets de las ómicas por separado, se pasan por el autoencoder y el bottleneck de ambos se une con un cbind()

Reordenar datasets

Vamos a utilizar los siguientes datasets:

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed y ExpProtTCGA_KIRC_RawData_w

Expresión génica

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_RowSort <- ExpGenTCGA_KIRC_Norm01_Filt
```

Expresión proteica

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort <- ExpProtTCGA_KIRC_RawDat
```

Juntar datasets

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01
> dim(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_
[1]  474 5074
```

Modelo 1: Concatenación + Autoencoder + Clasificador

Sampling: Creación de conjuntos de Test y Train

```
set.seed(231)
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01
```

Creación de etiquetas de test y train

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_SORTSample <- ExpProtTCGA_KIRC_RawData

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01
```

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01


ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01
```

Creación del encoder

```
enc_input_ExpGenExpProtNorm_M1 <- layer_input(shape = 5074)
enc_output_ExpGenExpProtNorm_M1 = enc_input_ExpGenExpProtNorm_M1 %>%
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units=20)

encoder_ExpGenExpProtNorm_M1 = keras_model(enc_input_ExpGenExpProtNorm_M1, enc_output_ExpGenExpProtNorm
> summary(encoder_ExpGenExpProtNorm_M1)
Model: "model_6"
____________________________________________________________________________________________________
Layer (type)                                  Output Shape                                 Param #
====================================================================================================
input_7 (InputLayer)                          [(None, 5074)]                               0
____________________________________________________________________________________________________
dense_12 (Dense)                              (None, 100)                                  507500
____________________________________________________________________________________________________
dense_11 (Dense)                              (None, 20)                                   2020
====================================================================================================
Total params: 509,520
Trainable params: 509,520
Non-trainable params: 0
____________________________________________________________________________________________________
```

Creación del decoder

```
dec_input_ExpGenExpProtNorm_M1 = layer_input(shape = 20)
dec_output_ExpGenExpProtNorm_M1 = dec_input_ExpGenExpProtNorm_M1 %>%
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units = 5074, activation = "relu")
decoder_ExpGenExpProtNorm_M1 = keras_model(dec_input_ExpGenExpProtNorm_M1, dec_output_ExpGenExpProtNorm
> summary(decoder_ExpGenExpProtNorm_M1)
Model: "model_7"
____________________________________________________________________________________________________
Layer (type)                                  Output Shape                                 Param #
====================================================================================================
input_8 (InputLayer)                          [(None, 20)]                                 0
____________________________________________________________________________________________________
dense_14 (Dense)                              (None, 100)                                  2100
____________________________________________________________________________________________________
dense_13 (Dense)                              (None, 5074)                                 512474
====================================================================================================
Total params: 514,574
```

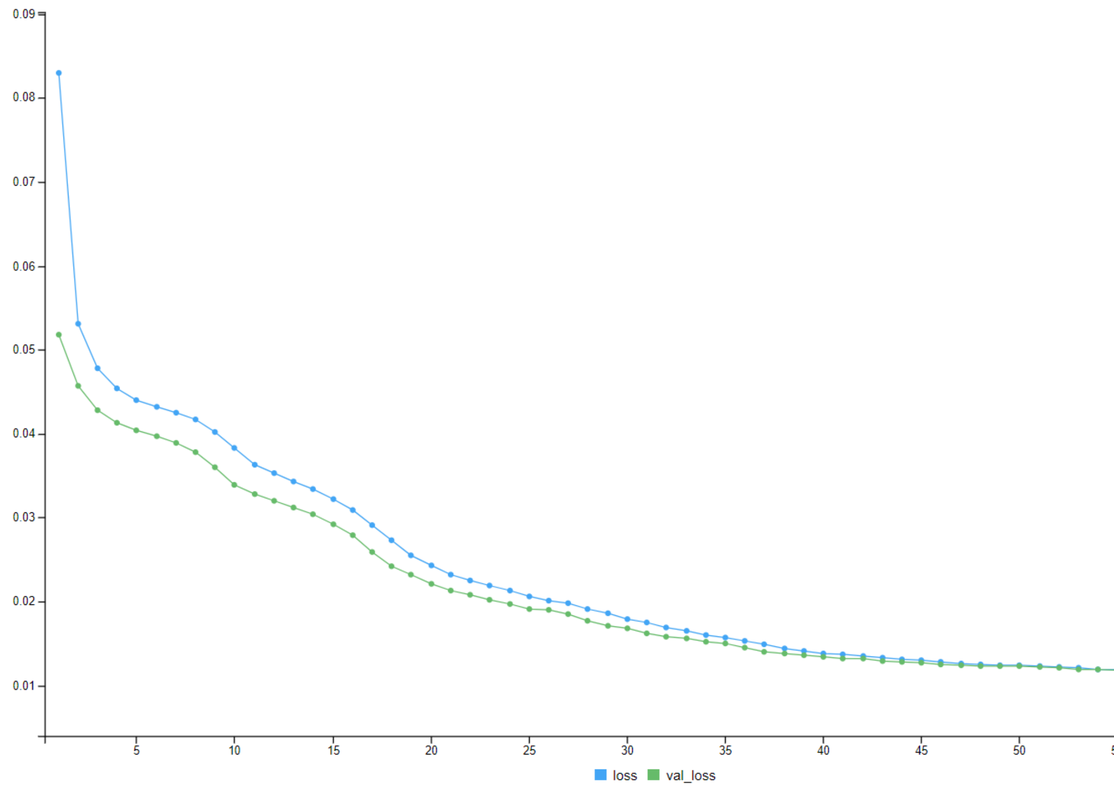
```
Trainable params: 514,574
Non-trainable params: 0
____________________________________________________________________________________________________
```

Definiendo el autoencoder

```
aen_input_ExpGenExpProtNorm_M1 = layer_input(shape = 5074)
aen_output_ExpGenExpProtNorm_M1 = aen_input_ExpGenExpProtNorm_M1 %>%
  encoder_ExpGenExpProtNorm_M1() %>%
  decoder_ExpGenExpProtNorm_M1()

aen_ExpGenExpProtNorm_M1 = keras_model(aen_input_ExpGenExpProtNorm_M1, aen_output_ExpGenExpProtNorm_M1)

> summary(aen_ExpGenExpProtNorm_M1)
Model: "model_2"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                                Param #
====================================================================================================
input_3 (InputLayer)                         [(None, 5074)]                              0
____________________________________________________________________________________________________
model (Model)                                (None, 20)                                  509520
____________________________________________________________________________________________________
model_1 (Model)                              (None, 5074)                                514574
====================================================================================================
Total params: 1,024,094
Trainable params: 1,024,094
Non-trainable params: 0
____________________________________________________________________________________________________

aen_ExpGenExpProtNorm_M1 %>% compile(optimizer="adam", loss="mean_squared_error")
```

Entrenamiento del modelo

```
history <- aen_ExpGenExpProtNorm_M1 %>% fit(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_T
```



Resultado autoencoder

```
> history

Final epoch (plot to see history):
    loss: 0.01106
val_loss: 0.01117
```

Vemos que el mse entre el input y el output es de 0.01117 para el conjunto test y 0.01106 para el conjunto train.

Obtener el bottleneck output del autoencodencoder

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm01
```

Guardado de datos

Conjuntos Train y Test

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N
```

Etiquetas

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_No

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_No
```

Clasificador con datos del bottleneck Haremos un script para utilizar el tfruns, asi será más sencillo encontrar el mejor modelo

Script: Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen

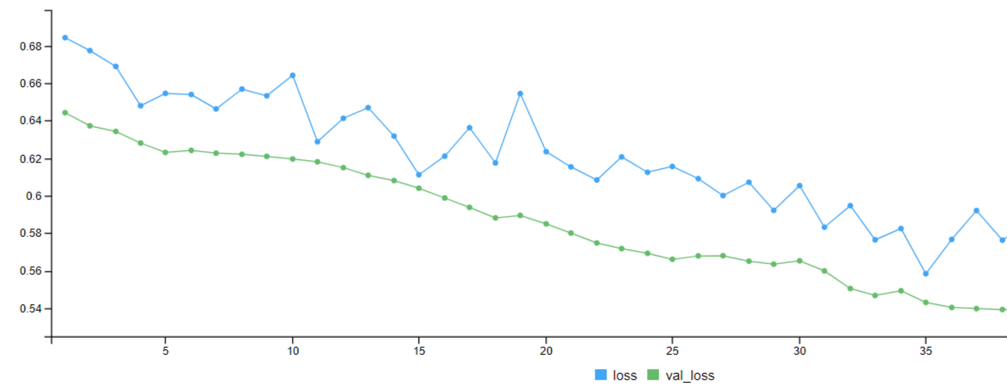
```
# test_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N

# train_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N

# test_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N

# train_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_N

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

# Definición del modelo Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen: muestras: 474 dimensiones: 

Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen <- keras_model_sequential() %>%
  layer_dense(units = 10, activation = "relu", input_shape = c(20)) %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 4, activation = "relu") %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 1, activation = "sigmoid")

Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen %>% fit(ExpProtTCGA_KIRC_RawData_woNA_Nor

plot(history)

score <- Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen %>% evaluate(
  ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_Norm
  verbose = 0
)

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

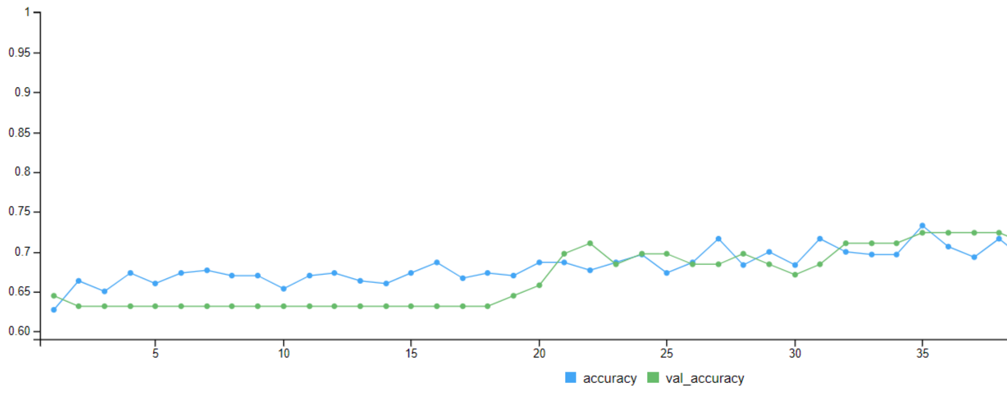
Resultados

Modelo 1

Resumen del modelo

```
> summary(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential"
________________________________________________________________________________________________
Layer (type)                                 Output Shape                            Param #
================================================================================================
dense_41 (Dense)                             (None, 10)                              210
________________________________________________________________________________________________
dropout_1 (Dropout)                          (None, 10)                              0
________________________________________________________________________________________________
dense_40 (Dense)                             (None, 4)                               44
________________________________________________________________________________________________
dropout (Dropout)                            (None, 4)                               0
________________________________________________________________________________________________
dense_39 (Dense)                             (None, 1)                               5
================================================================================================
Total params: 259
Trainable params: 259
Non-trainable params: 0
________________________________________________________________________________________________
```





Gráficas de pérdida y precisión

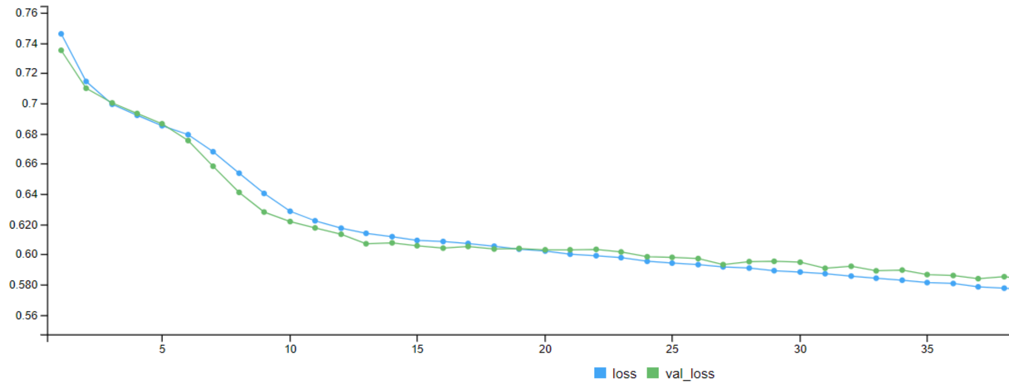
```
> history

Final epoch (plot to see history):
        loss: 0.5582
    accuracy: 0.7228
    val_loss: 0.5392
val_accuracy: 0.7237
> score
     loss  accuracy
0.6371682 0.6000000
```

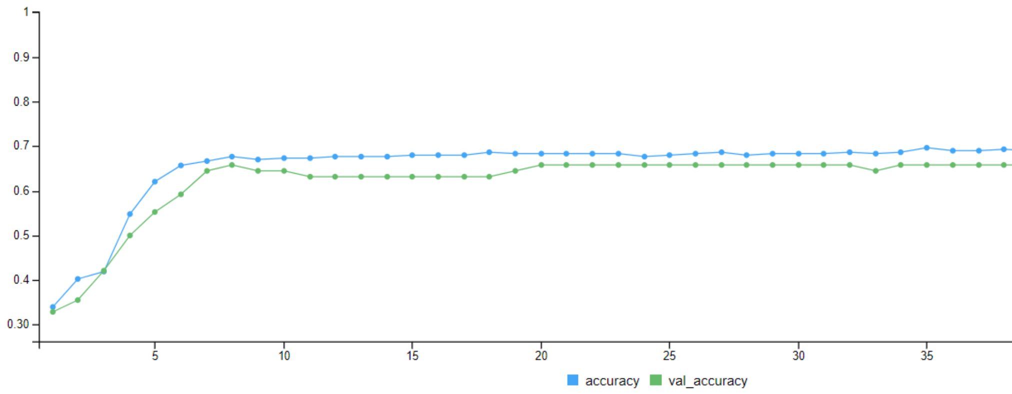
Modelo 2

Resumen del modelo

```
> summary(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_1"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                             Param #
====================================================================================================
dense_44 (Dense)                             (None, 10)                               210
____________________________________________________________________________________________________
dense_43 (Dense)                             (None, 4)                                44
____________________________________________________________________________________________________
dense_42 (Dense)                             (None, 1)                                5
====================================================================================================
Total params: 259
```

```
Trainable params: 259
Non-trainable params: 0
________________________________________________________________________________________________________
```





Gráficas de pérdida y precisión

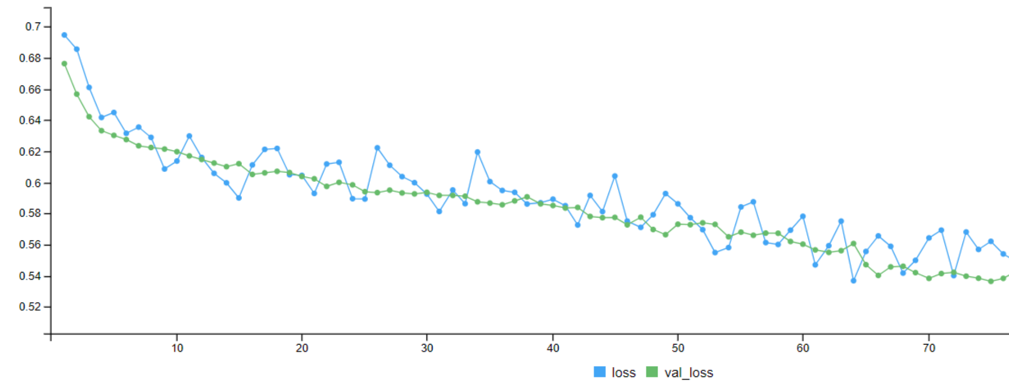
```
> score
     loss  accuracy
0.6191577 0.6842105
> history

Final epoch (plot to see history):
        loss: 0.5648
    accuracy: 0.6964
    val_loss: 0.5737
val_accuracy: 0.6974
```

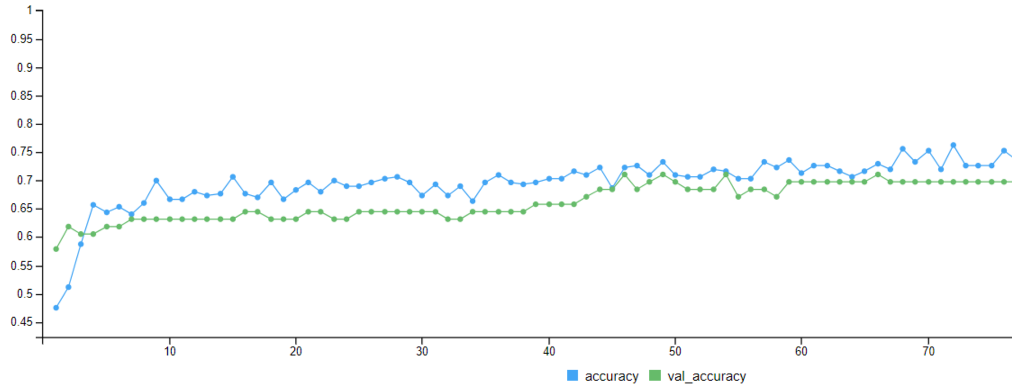
Modelo 3

Resumen del modelo

```
> summary(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_6"
________________________________________________________________________________________________________
Layer (type)                                  Output Shape                                    Param #
========================================================================================================
dense_62 (Dense)                              (None, 16)                                      336
________________________________________________________________________________________________________
```

```
dropout_6 (Dropout)                          (None, 16)                             0
____________________________________________________________________________________________________
dense_61 (Dense)                             (None, 8)                              136
____________________________________________________________________________________________________
dropout_5 (Dropout)                          (None, 8)                              0
____________________________________________________________________________________________________
dense_60 (Dense)                             (None, 4)                              36
____________________________________________________________________________________________________
dense_59 (Dense)                             (None, 1)                              5
====================================================================================================
Total params: 513
Trainable params: 513
Non-trainable params: 0
____________________________________________________________________________________________________
```





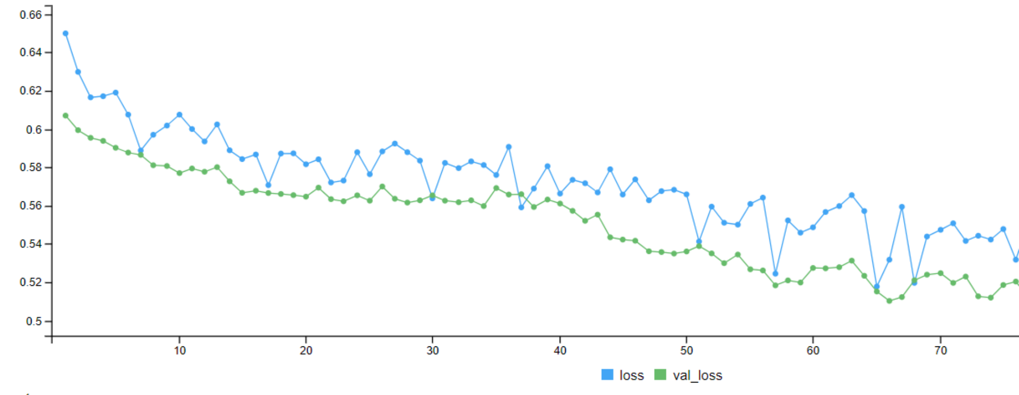
Gráficas de pérdida y precisión

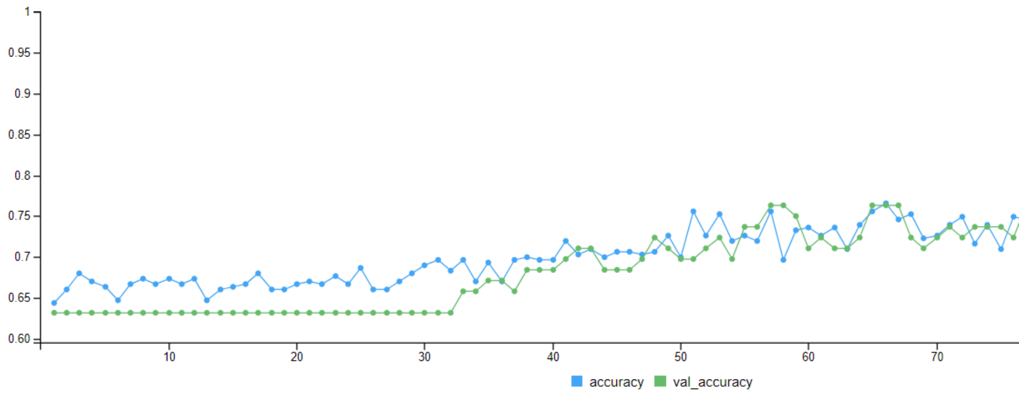
```
> score
     loss  accuracy
0.6193671 0.6736842
> history

Final epoch (plot to see history):
        loss: 0.5376
    accuracy: 0.7492
    val_loss: 0.537
val_accuracy: 0.6974
```

Modelo 4

Resumen del modelo

```
> summary(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_12"
________________________________________________________________________________________________________
Layer (type)                               Output Shape                            Param #
========================================================================================================
dense_88 (Dense)                           (None, 18)                              378
________________________________________________________________________________________________________
dropout_16 (Dropout)                       (None, 18)                              0
________________________________________________________________________________________________________
dense_87 (Dense)                           (None, 8)                               152
________________________________________________________________________________________________________
dense_86 (Dense)                           (None, 5)                               45
________________________________________________________________________________________________________
dense_85 (Dense)                           (None, 1)                               6
========================================================================================================
Total params: 581
Trainable params: 581
Non-trainable params: 0
________________________________________________________________________________________________________
```





Gráficas de pérdida y precisión

```
> score
     loss  accuracy
0.6462954 0.6631579
> history
```

```
Final epoch (plot to see history):
        loss: 0.5299
    accuracy: 0.7624
    val_loss: 0.5198
val_accuracy: 0.7237
```

SVM del modelo La precisión de este modelo como SVM es de 66.31%.

```
Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM <- cbind(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameS
colnames(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM)[1] <- "Labels"

# Entrenamiento del modelo
Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR = svm(formula = Labels ~ .,
                 data = Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')
> str(Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR)
List of 30
 $ call          : language svm(formula = Labels ~ ., data = Clasificador_Modelo1Autoencoder_ExpProt_A
 $ type          : num 0
 $ kernel        : num 2
 $ cost          : num 1
 $ degree        : num 3
 $ gamma         : num 0.05
 $ coef0         : num 0
 $ nu            : num 0.5
 $ epsilon       : num 0.1
 $ sparse        : logi FALSE
 $ scaled        : logi [1:20] TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ x.scale       :List of 2
  ..$ scaled:center: Named num [1:20] -2.741 0.976 0.107 0.665 2.648 ...
  .. ..- attr(*, "names")= chr [1:20] "V2" "V3" "V4" "V5" ...
  ..$ scaled:scale : Named num [1:20] 1.098 0.573 0.552 1.047 1.118 ...
  .. ..- attr(*, "names")= chr [1:20] "V2" "V3" "V4" "V5" ...
 $ y.scale       : NULL
 $ nclasses      : int 2
 $ levels        : chr [1:2] "0" "1"
 $ tot.nSV       : int 270
 $ nSV           : int [1:2] 149 121
 $ labels        : int [1:2] 2 1
 $ SV            : num [1:270, 1:20] 1.483 1.631 0.449 1.098 -4.437 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:270] "1" "2" "3" "4" ...
  .. ..$ : chr [1:20] "V2" "V3" "V4" "V5" ...
 $ index         : int [1:270] 1 2 3 4 9 10 14 16 18 20 ...
 $ rho           : num -0.336
 $ compprob      : logi FALSE
 $ probA         : NULL
 $ probB         : NULL
 $ sigma         : NULL
 $ coefs         : num [1:270, 1] 1 1 1 0.753 0.677 ...
 $ na.action     : NULL
 $ fitted        : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 1 2 2 2 ...
```

```
  ..- attr(*, "names")= chr [1:379] "1" "2" "3" "4" ...
 $ decision.values: num [1:379, 1] 0.5448 0.0134 0.9554 0.9998 1.1077 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:379] "1" "2" "3" "4" ...
  .. ..$ : chr "1/0"
 $ terms          :Classes 'terms', 'formula'  language Labels ~ V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 +
  .. ..- attr(*, "variables")= language list(Labels, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13
  .. ..- attr(*, "factors")= int [1:21, 1:20] 0 1 0 0 0 0 0 0 0 0 ...
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:21] "Labels" "V2" "V3" "V4" ...
  .. .. .. ..$ : chr [1:20] "V2" "V3" "V4" "V5" ...
  .. ..- attr(*, "term.labels")= chr [1:20] "V2" "V3" "V4" "V5" ...
  .. ..- attr(*, "order")= int [1:20] 1 1 1 1 1 1 1 1 1 1 ...
  .. ..- attr(*, "intercept")= num 0
  .. ..- attr(*, "response")= int 1
  .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
  .. ..- attr(*, "predvars")= language list(Labels, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13,
  .. ..- attr(*, "dataClasses")= Named chr [1:21] "numeric" "numeric" "numeric" "numeric" ...
  .. .. ..- attr(*, "names")= chr [1:21] "Labels" "V2" "V3" "V4" ...
 - attr(*, "class")= chr [1:2] "svm.formula" "svm"

# Predicciones
Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR = predict(Clasificador_Modelo1A

# Matriz de confusión

cmR_Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR = table(Clasificador_Model
> cmR_Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR
                                                                           ExpProtTCGA_KIRC_RawData_woNA_N
Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR  0  1
                                                                        0 13  9
                                                                        1 23 50
> (50+13)/(50+13+23+9)
[1] 0.6631579
```

Modelo 2: Autoencoder + Concatenación + Clasificador

Tratamiento dataset Expresión génica x_train (Train de Expresión génica con las mismas muestras que Expresión proteica) ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Train

x_test (Test de Expresión génica con las mismas muestras que Expresión proteica) ExpGenTCGA_KIRC_Norm01_Filt75_Sa

y_train (Labels Train de Expresión génica con las mismas muestras que Expresión proteica) ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Train_FactorNumb

y_test (Labels Test de Expresión génica con las mismas muestras que Expresión proteica) ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Labels_Test_FactorNumb

Creación de encoder

```
enc_input_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 <- layer_input(shape = 4897)
enc_output_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = enc_input_ExpGenTCGA_KIRC_Norm01_Filt75_Sa
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units=20)
```

```
encoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = keras_model(enc_input_ExpGenTCGA_KIRC_Norm01_

> summary(encoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2)
Model: "model_24"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                           Param #
====================================================================================================
input_26 (InputLayer)                        [(None, 4897)]                         0
____________________________________________________________________________________________________
dense_90 (Dense)                             (None, 100)                            489800
____________________________________________________________________________________________________
dense_89 (Dense)                             (None, 20)                             2020
====================================================================================================
Total params: 491,820
Trainable params: 491,820
Non-trainable params: 0
____________________________________________________________________________________________________
```

Creación de decoder

```
dec_input_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = layer_input(shape = 20)
dec_output_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = dec_input_ExpGenTCGA_KIRC_Norm01_Filt75_Sa
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units = 4897, activation = "relu")
decoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = keras_model(dec_input_ExpGenTCGA_KIRC_Norm01_
> summary(decoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2)
Model: "model_26"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                           Param #
====================================================================================================
input_28 (InputLayer)                        [(None, 20)]                           0
____________________________________________________________________________________________________
dense_94 (Dense)                             (None, 100)                            2100
____________________________________________________________________________________________________
dense_93 (Dense)                             (None, 4897)                           494597
====================================================================================================
Total params: 496,697
Trainable params: 496,697
Non-trainable params: 0
____________________________________________________________________________________________________
```

Definiendo el autoencoder

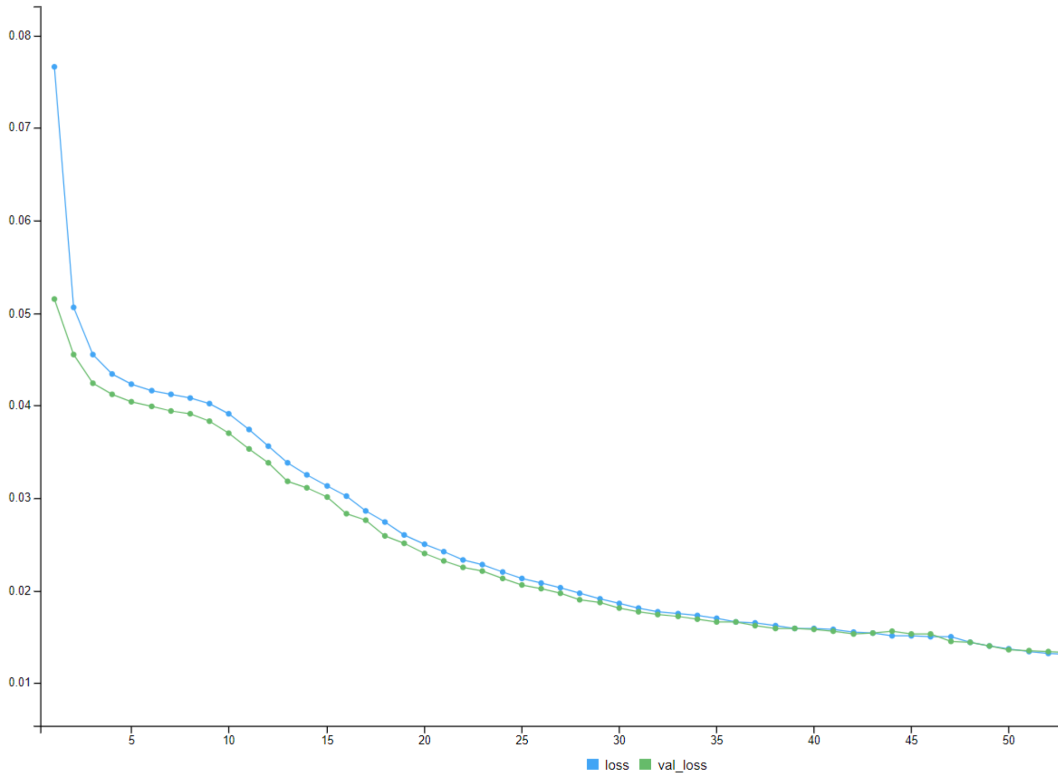
```
aen_input_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = layer_input(shape = 4897)
aen_output_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = aen_input_ExpGenTCGA_KIRC_Norm01_Filt75_Sa
  encoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2() %>%
  decoder_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2()

aen_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 = keras_model(aen_input_ExpGenTCGA_KIRC_Norm01_Fil

> summary(aen_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2)
Model: "model_27"
```

```
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                             Param #
====================================================================================================
input_29 (InputLayer)                        [(None, 4897)]                           0
____________________________________________________________________________________________________
model_24 (Model)                             (None, 20)                               491820
____________________________________________________________________________________________________
model_26 (Model)                             (None, 4897)                             496697
====================================================================================================
Total params: 988,517
Trainable params: 988,517
Non-trainable params: 0
____________________________________________________________________________________________________
```

```
aen_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 %>% compile(optimizer="adam", loss="mean_squared_e
```

Entrenamiento del modelo

```
history <- aen_ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_M2 %>% fit(ExpGenTCGA_KIRC_Norm01_Filt75_Sa
```



Resultado del autoencoder

```
> history

Final epoch (plot to see history):
    loss: 0.01184
val_loss: 0.01225
```

Obtener el bottleneck output del autoencoder

```
ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Bottleneck <- predict(encoder_ExpGenTC
```

Tratamiento dataset Expresión proteica/proteómica x_train (Train de Expresión proteica con las mismas muestras que Expresión génica) ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Tr

x_test (Test de Expresión proteica con las mismas muestras que Expresión génica) ExpProtTCGA_KIRC_RawData_woNA_

y_train (Labels Train de Expresión proteica con las mismas muestras que Expresión génica) ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Train_FactorNumb

y_test (Labels Test de Expresión proteica con las mismas muestras que Expresión génica) ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Labels_Test_FactorNumb

Creación de encoder

```
enc_input_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 <- layer_input(shape = 177)
enc_output_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = enc_input_ExpProtTCGA_KIRC_RawData_wol
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units=20)

encoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = keras_model(enc_input_ExpProtTCGA_KIRC_R

> summary(encoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2)
Model: "model_32"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                              Param #
====================================================================================================
input_34 (InputLayer)                        [(None, 177)]                             0
____________________________________________________________________________________________________
dense_102 (Dense)                            (None, 100)                               17800
____________________________________________________________________________________________________
dense_101 (Dense)                            (None, 20)                                2020
====================================================================================================
Total params: 19,820
Trainable params: 19,820
Non-trainable params: 0
____________________________________________________________________________________________________
```

Creación de decoder

```
dec_input_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = layer_input(shape = 20)
dec_output_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = dec_input_ExpProtTCGA_KIRC_RawData_wol
  layer_dense(units=100, activation = "relu") %>%
  layer_dense(units = 177, activation = "relu")
decoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = keras_model(dec_input_ExpProtTCGA_KIRC_R
> summary(decoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2)
Model: "model_33"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                              Param #
====================================================================================================
input_35 (InputLayer)                        [(None, 20)]                              0
```

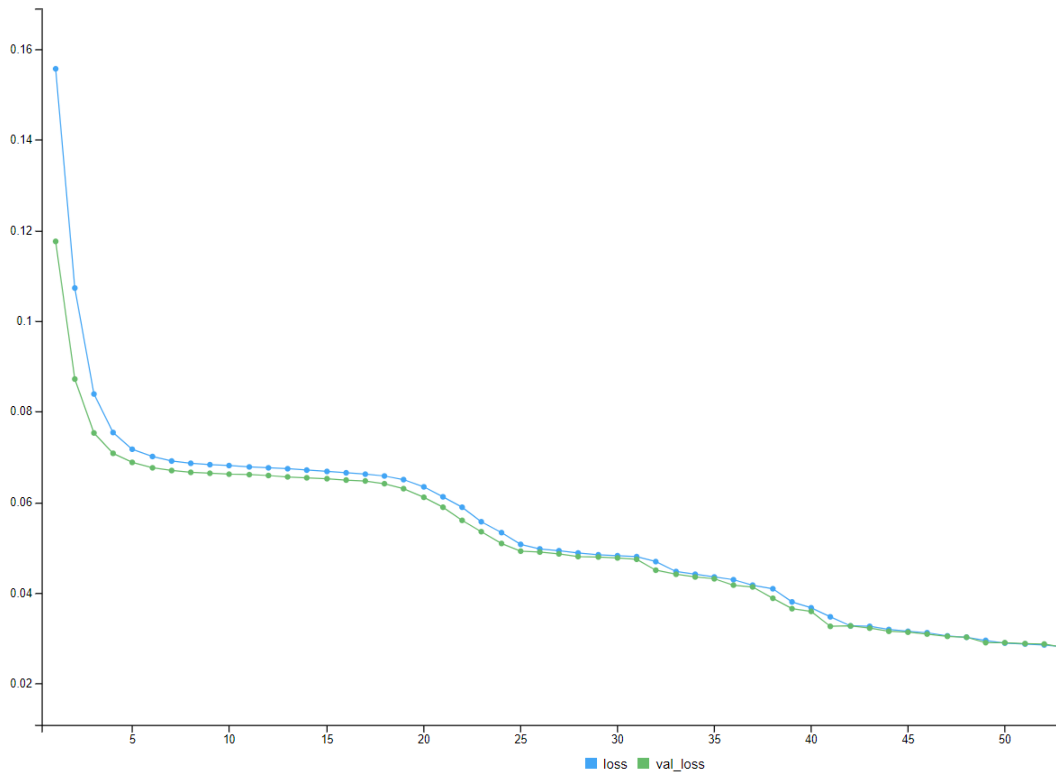
```
____________________________________________________________________________________________________
dense_104 (Dense)                            (None, 100)                                    2100
____________________________________________________________________________________________________
dense_103 (Dense)                            (None, 177)                                    17877
====================================================================================================
Total params: 19,977
Trainable params: 19,977
Non-trainable params: 0
____________________________________________________________________________________________________
```

Definiendo el autoencoder

```
aen_input_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = layer_input(shape = 177)
aen_output_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = aen_input_ExpProtTCGA_KIRC_RawData_wo
  encoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2() %>%
  decoder_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2()

aen_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 = keras_model(aen_input_ExpProtTCGA_KIRC_RawDa

>  summary(aen_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2)
Model: "model_34"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                                   Param #
====================================================================================================
input_36 (InputLayer)                        [(None, 177)]                                  0
____________________________________________________________________________________________________
model_32 (Model)                             (None, 20)                                     19820
____________________________________________________________________________________________________
model_33 (Model)                             (None, 177)                                    19977
====================================================================================================
Total params: 39,797
Trainable params: 39,797
Non-trainable params: 0
____________________________________________________________________________________________________

aen_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 %>% compile(optimizer="adam", loss="mean_squar
```

Entrenamiento del modelo

```
history <- aen_ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_M2 %>% fit(ExpProtTCGA_KIRC_RawData_wo
```



Resultado del autoencoder

```
> history

Final epoch (plot to see history):
    loss: 0.02438
val_loss: 0.02391
```

Obtener el bottleneck output del autoencoder

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck <- predict(encoder_ExpPr
```

Concatenación de datasets

Ordenar datasets

```
# Expresión génica

ExpGenTCGA_KIRC_Norm01_Filt75_SameSampExpProt_Renamed_Transposed_Bottleneck_RowSort <- ExpGenTCGA_KIRC_N

# Proteomica

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort <- ExpProtTCGA_K
```

Concatenar

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGenTCGA_K
```

Sampling: Creación de conjuntos de Test y Train

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGenTCGA_
```

```
ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGenTCGA_
```

Las etiquetas sirven las mismas que en modelo anterior:

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTC

ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTC

Guardado de datos

```
# Train
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGen

# Test
save(ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGen
```

Clasificador con datos del bottleneck Modelo 2 Haremos un script para utilizar el tfruns, asi será más sencillo encontrar el mejor modelo

Script: Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen

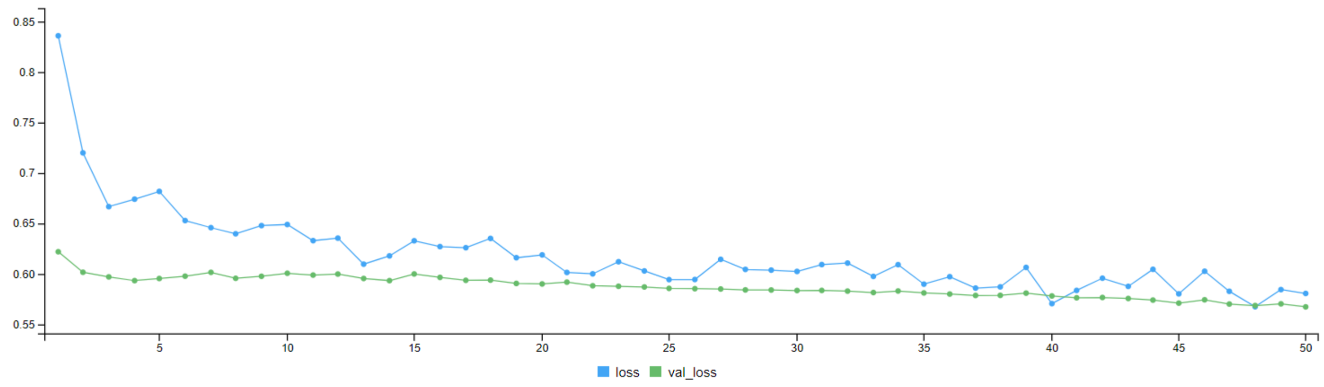
```
# test_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGen

# train_x
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGen

# test_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_

# train_y
load("ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_RowSort_AND_ExpGenTCGA_KIRC_

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

# Definición del modelo Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen: muestras: 474 dimensiones: 

Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen <- keras_model_sequential() %>%
  layer_dense(units = 10, activation = "relu", input_shape = c(40)) %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 4, activation = "relu") %>%
  layer_dropout(0.2) %>%
  layer_dense(units = 1, activation = "sigmoid")

Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen %>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("accuracy")
)

# Entrenamiento y evaluación del modelo

history <- Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen %>% fit(ExpProtTCGA_KIRC_RawData_woNA_Nor

plot(history)

score <- Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen %>% evaluate(
  ExpProtTCGA_KIRC_RawData_woNA_Norm_SameSampExpGen_Renamed_Transposed_Bottleneck_RowSort_AND_ExpGenTCGA
  verbose = 0
)

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

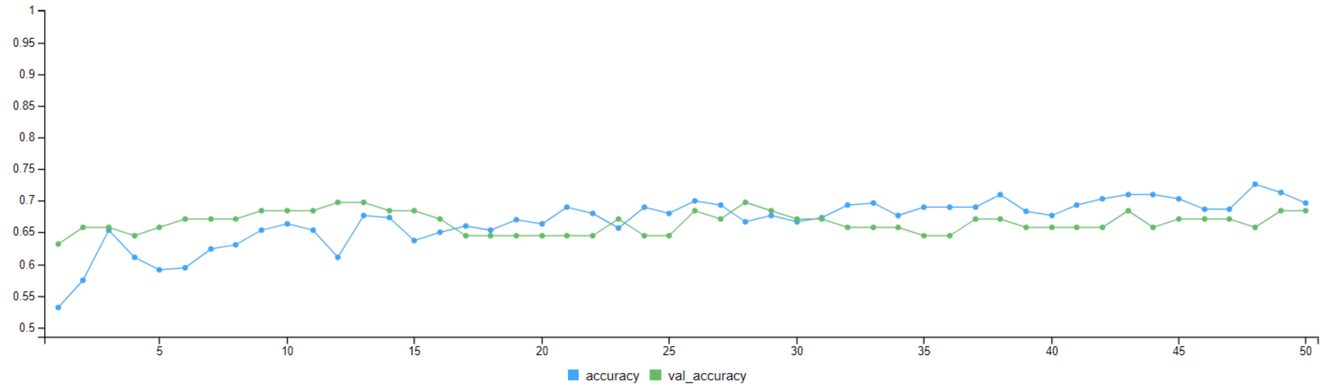
Resultados

Modelo 1 Resumen del modelo

```
> summary(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_13"
________________________________________________________________________________________________________
Layer (type)                                 Output Shape                             Param #
========================================================================================================
dense_107 (Dense)                            (None, 10)                               410
________________________________________________________________________________________________________
dropout_18 (Dropout)                         (None, 10)                               0
________________________________________________________________________________________________________
dense_106 (Dense)                            (None, 4)                                44
________________________________________________________________________________________________________
dropout_17 (Dropout)                         (None, 4)                                0
________________________________________________________________________________________________________
dense_105 (Dense)                            (None, 1)                                5
========================================================================================================
Total params: 459
Trainable params: 459
Non-trainable params: 0
________________________________________________________________________________________________________
```

Precisión y loss



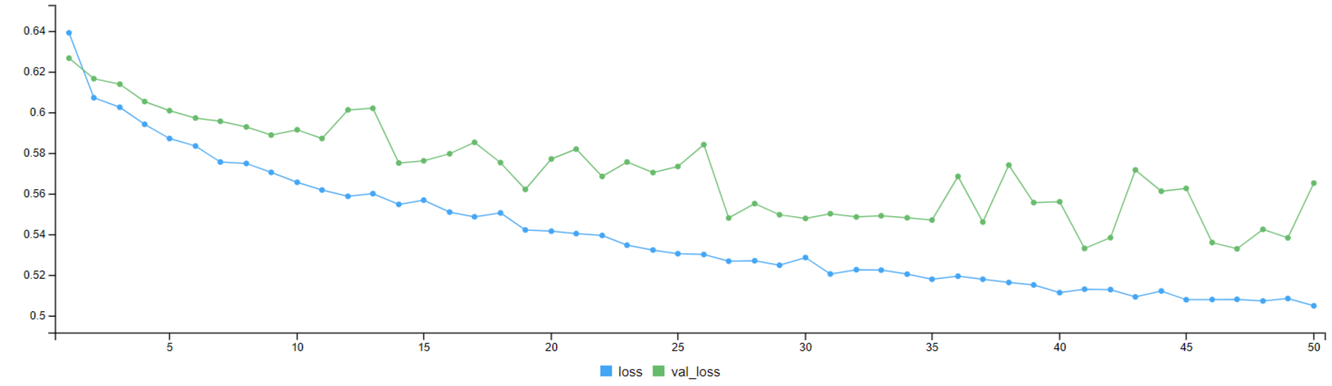


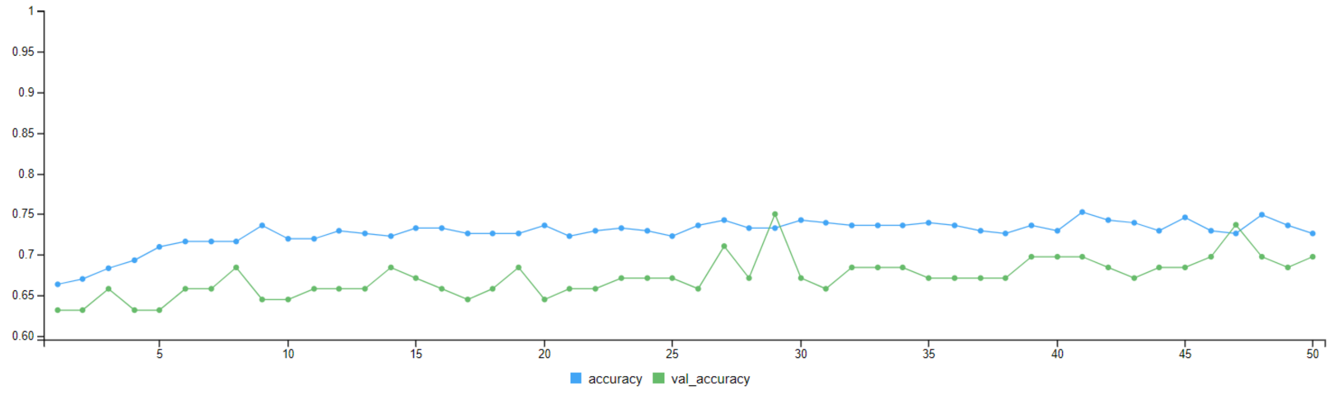
```
> score
     loss  accuracy
0.6364123 0.6210526
> history

Final epoch (plot to see history):
        loss: 0.581
    accuracy: 0.6964
    val_loss: 0.5677
val_accuracy: 0.6842
```

Modelo 2 Resumen del modelo

```
> summary(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_14"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                            Param #
====================================================================================================
dense_110 (Dense)                            (None, 20)                              820
____________________________________________________________________________________________________
dense_109 (Dense)                            (None, 8)                               168
____________________________________________________________________________________________________
dense_108 (Dense)                            (None, 1)                               9
====================================================================================================
Total params: 997
Trainable params: 997
Non-trainable params: 0
____________________________________________________________________________________________________
```

Precisión y loss





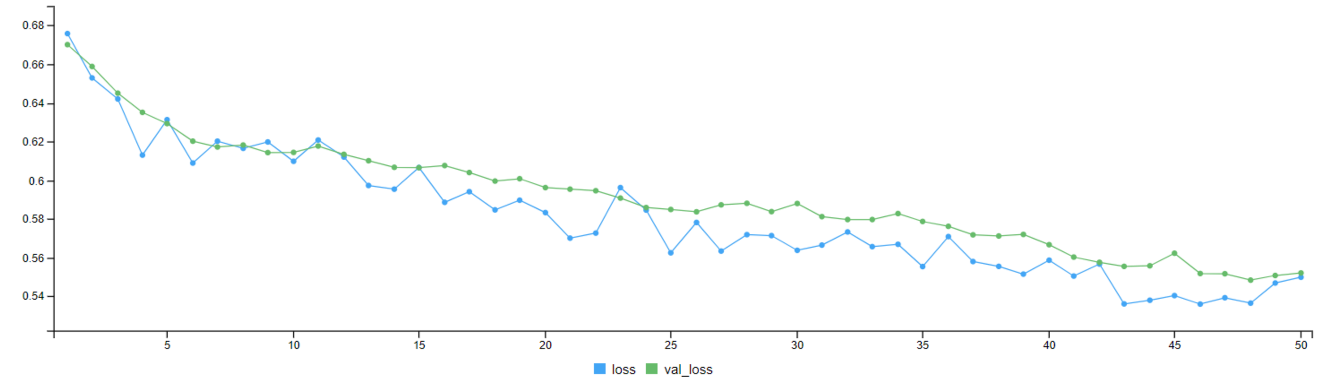
```
> score
     loss  accuracy
0.6747756 0.6315789
> history

Final epoch (plot to see history):
        loss: 0.5046
    accuracy: 0.7261
    val_loss: 0.5651
val_accuracy: 0.6974
```

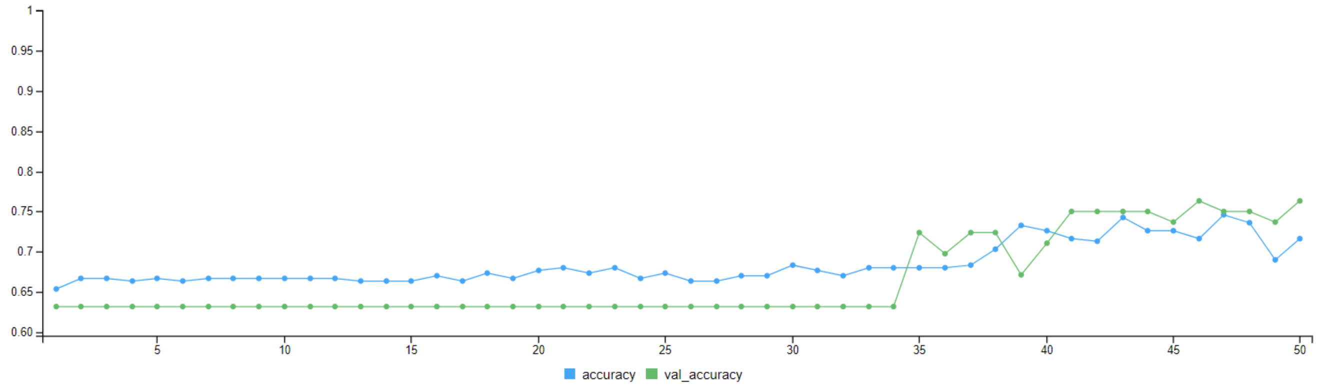
Modelo 3 Resumen del modelo

```
> summary(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_15"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                           Param #
====================================================================================================
dense_114 (Dense)                            (None, 20)                             820
____________________________________________________________________________________________________
dropout_20 (Dropout)                         (None, 20)                             0
____________________________________________________________________________________________________
dense_113 (Dense)                            (None, 16)                             336
____________________________________________________________________________________________________
dropout_19 (Dropout)                         (None, 16)                             0
____________________________________________________________________________________________________
```

```
dense_112 (Dense)                    (None, 8)                        136
________________________________________________________________________________
dense_111 (Dense)                    (None, 1)                        9
================================================================================
Total params: 1,301
Trainable params: 1,301
Non-trainable params: 0
________________________________________________________________________________
```

Precisión y loss





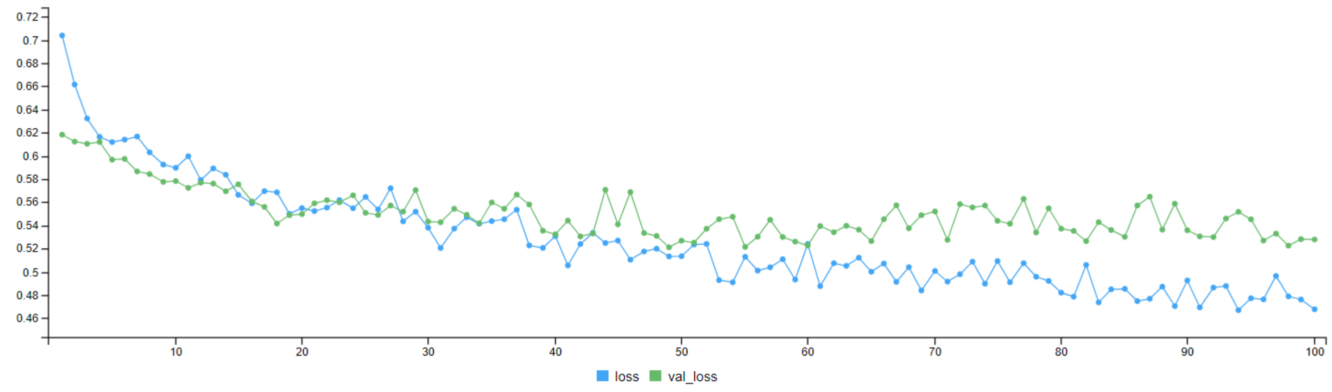
```
> score
     loss  accuracy
0.6102709 0.6947368
> history

Final epoch (plot to see history):
        loss: 0.5499
    accuracy: 0.7162
    val_loss: 0.5521
val_accuracy: 0.7632
```

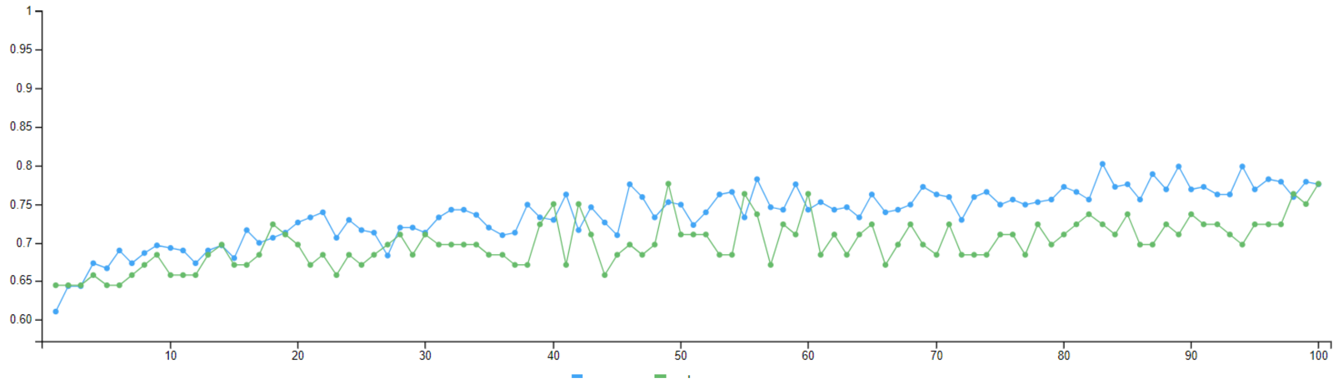
Modelo 4 Resumen del modelo

```
> summary(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_31"
________________________________________________________________________________
Layer (type)                         Output Shape                     Param #
```

```
================================================================================================
dense_175 (Dense)                           (None, 35)                            1435
________________________________________________________________________________________________
dropout_44 (Dropout)                        (None, 35)                            0
________________________________________________________________________________________________
dense_174 (Dense)                           (None, 20)                            720
________________________________________________________________________________________________
dropout_43 (Dropout)                        (None, 20)                            0
________________________________________________________________________________________________
dense_173 (Dense)                           (None, 10)                            210
________________________________________________________________________________________________
dense_172 (Dense)                           (None, 1)                             11
================================================================================================
Total params: 2,376
Trainable params: 2,376
Non-trainable params: 0
________________________________________________________________________________________________
```

Precisión y loss





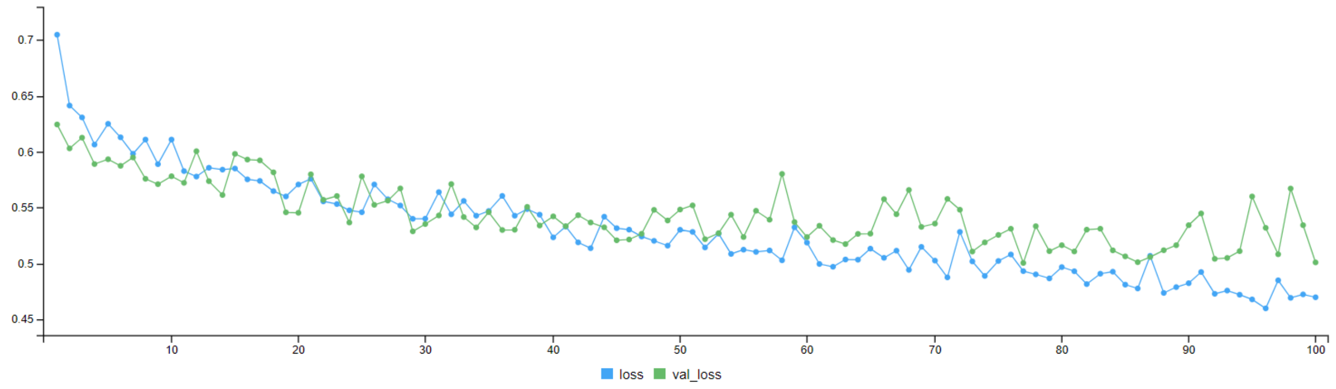
```
> score
     loss  accuracy
0.6541336 0.6736842
> history

Final epoch (plot to see history):
        loss: 0.4678
    accuracy: 0.7756
    val_loss: 0.5279
```

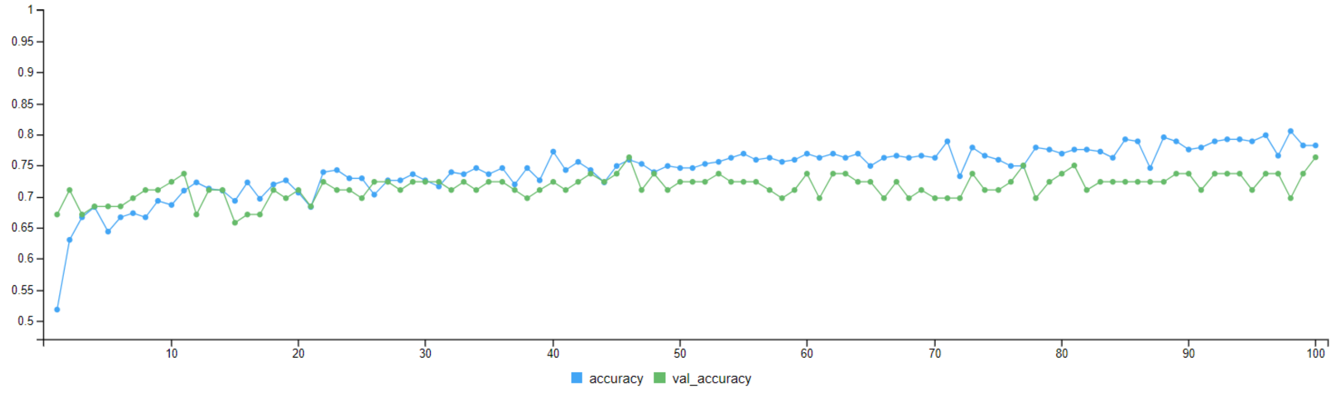
```
val_accuracy: 0.7763
```

Modelo 5 Resumen del modelo

```
> summary(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen)
Model: "sequential_71"
____________________________________________________________________________________________________
Layer (type)                                 Output Shape                                Param #
====================================================================================================
dense_348 (Dense)                            (None, 40)                                  1640
____________________________________________________________________________________________________
dropout_112 (Dropout)                        (None, 40)                                  0
____________________________________________________________________________________________________
dense_347 (Dense)                            (None, 20)                                  820
____________________________________________________________________________________________________
dense_346 (Dense)                            (None, 8)                                   168
____________________________________________________________________________________________________
dense_345 (Dense)                            (None, 4)                                   36
____________________________________________________________________________________________________
dense_344 (Dense)                            (None, 1)                                   5
====================================================================================================
Total params: 2,669
Trainable params: 2,669
Non-trainable params: 0
____________________________________________________________________________________________________
```

Precisión y loss





```
> score
     loss  accuracy
0.6559734 0.7263158
> history

Final epoch (plot to see history):
        loss: 0.4699
    accuracy: 0.7822
    val_loss: 0.5011
val_accuracy: 0.7632
```

SVM del modelo La precisión de este modelo como SVM es de 65.26%.

```
Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM <- cbind(ExpProtTCGA_KIRC_RawData_woNA_Norm_Same
colnames(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM)[1] <- "Labels"
```

Entrenamiento del modelo

```
Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR = svm(formula = Labels ~ .,
                 data = Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM,
                 type = 'C-classification', # this is because we want to make a regression classificati
                 kernel = 'radial')
> str(Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR)
List of 30
 $ call           : language svm(formula = Labels ~ ., data = Clasificador_Modelo2Autoencoder_ExpProt_A
 $ type           : num 0
 $ kernel         : num 2
 $ cost           : num 1
 $ degree         : num 3
 $ gamma          : num 0.025
 $ coef0          : num 0
 $ nu             : num 0.5
 $ epsilon        : num 0.1
 $ sparse         : logi FALSE
 $ scaled         : logi [1:40] TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ x.scale        :List of 2
  ..$ scaled:center: Named num [1:40] 0.838 0.257 -0.601 -1.483 -0.575 ...
  .. ..- attr(*, "names")= chr [1:40] "V2" "V3" "V4" "V5" ...
  ..$ scaled:scale : Named num [1:40] 0.244 0.194 0.299 0.165 0.162 ...
  .. ..- attr(*, "names")= chr [1:40] "V2" "V3" "V4" "V5" ...
 $ y.scale        : NULL
 $ nclasses       : int 2
 $ levels         : chr [1:2] "0" "1"
 $ tot.nSV        : int 284
 $ nSV            : int [1:2] 161 123
 $ labels         : int [1:2] 2 1
 $ SV             : num [1:284, 1:40] -1.134 -0.669 -1.292 -0.408 -1.835 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:284] "1" "2" "3" "4" ...
  .. ..$ : chr [1:40] "V2" "V3" "V4" "V5" ...
 $ index          : int [1:284] 1 2 3 4 5 9 10 14 16 18 ...
 $ rho            : num -0.351
 $ compprob       : logi FALSE
 $ probA          : NULL
 $ probB          : NULL
 $ sigma          : NULL
 $ coefs          : num [1:284, 1] 1 1 0.674 1 0.565 ...
 $ na.action      : NULL
 $ fitted         : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 1 2 2 2 ...
  ..- attr(*, "names")= chr [1:379] "1" "2" "3" "4" ...
 $ decision.values: num [1:379, 1] 0.906 0.219 1 0.534 1 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:379] "1" "2" "3" "4" ...
  .. ..$ : chr "1/0"
 $ terms          :Classes 'terms', 'formula'  language Labels ~ V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 +
  .. ..- attr(*, "variables")= language list(Labels, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13
  .. ..- attr(*, "factors")= int [1:41, 1:40] 0 1 0 0 0 0 0 0 0 0 ...
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:41] "Labels" "V2" "V3" "V4" ...
  .. .. .. ..$ : chr [1:40] "V2" "V3" "V4" "V5" ...
  .. ..- attr(*, "term.labels")= chr [1:40] "V2" "V3" "V4" "V5" ...
  .. ..- attr(*, "order")= int [1:40] 1 1 1 1 1 1 1 1 1 1 ...
  .. ..- attr(*, "intercept")= num 0
  .. ..- attr(*, "response")= int 1
  .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
  .. ..- attr(*, "predvars")= language list(Labels, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13,
  .. ..- attr(*, "dataClasses")= Named chr [1:41] "numeric" "numeric" "numeric" "numeric" ...
  .. .. ..- attr(*, "names")= chr [1:41] "Labels" "V2" "V3" "V4" ...
 - attr(*, "class")= chr [1:2] "svm.formula" "svm"

# Predicciones
Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR = predict(Clasificador_Modelo2A

# Matriz de confusión

cmR_Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR = table(Clasificador_Modelo
> cmR_Clasificador_Modelo2Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR
                                                                       ExpProtTCGA_KIRC_RawData_woNA_No
Clasificador_Modelo1Autoencoder_ExpProt_AND_ExpGen_SVM_classifierR_predR  0  1
                                                                       0 13  9
                                                                       1 23 50

> (53+9)/(53+9+27+6)
[1] 0.6526316
```