Clasificador: prueba no TCGA con muestras de Control 10000 sondas

Alberto Joven Álvarez

26 de noviembre, 2022

Índice

1 Lista de librerías empleadas

```
library(knitr)
library(dplyr)
library(readr)
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)
library(FDb.InfiniumMethylation.hg19)
library(impute)
library(SummarizedExperiment)
library(GEOquery)
library(caret)
library(ggplot2)
library(keras)
```

2 Entrenamiento del modelo red neuronal con 1.000 sondas más variables.

2.1 Carga de los datos: obteto summarizedExperiment y desglose por tipo de sonda

Se han seleccionado las 10000 sondas más variables, una vez excluidas sondas problemáticas

```
load("G:/TFM UOC/datos/Clasificador_variabilidad/sondas_10000_all.Rda")
data_10000
```

```
## class: RangedSummarizedExperiment
## dim: 10000 9707
## metadata(0):
## assays(1): counts
## rownames(10000): cg20148575 cg04039555 ... cg02936049 cg08628006
## rowData names(10): addressA addressB ... probeEnd probeTarget
## colnames(9707): TCGA-2F-A9KO-01A-11D-A38H-05
##   TCGA-2F-A9KP-01A-11D-A38H-05 ... TCGA-ZA-A8F6-01A-23D-A365-05
##   TCGA-ZQ-A9CR-01A-11D-A398-05
## colData names(7): bar_code sujeto ... assay label
```

```
sondas_10000 <- data_10000

codigos <- c("01", "03", "11")
sondas_10000 <- sondas_10000[ , sondas_10000$categoria %in% codigos]

train <- assay(sondas_10000, "counts") %>%  t()
label <- sondas_10000$label %>% factor()
label_c <- to_categorical(as.integer(label))
```

```
## Loaded Tensorflow version 2.9.2
```

```
tipos_sondas <- rowData(sondas_10000)$channel %>%  as.factor()
train_green_data <- train[ , tipos_sondas=="Grn"]
train_red_data <- train[ , tipos_sondas == "Red"]
train_both_data <- train[ , tipos_sondas =="Both"]

sondas_10000
```

```
## class: RangedSummarizedExperiment
## dim: 10000 9267
## metadata(0):
## assays(1): counts
## rownames(10000): cg20148575 cg04039555 ... cg02936049 cg08628006
## rowData names(10): addressA addressB ... probeEnd probeTarget
## colnames(9267): TCGA-2F-A9KO-01A-11D-A38H-05
##   TCGA-2F-A9KP-01A-11D-A38H-05 ... TCGA-ZA-A8F6-01A-23D-A365-05
##   TCGA-ZQ-A9CR-01A-11D-A398-05
## colData names(7): bar_code sujeto ... assay label
```

2.2 El modelo

```
build_model <- function() {
  green <- layer_input(shape= dim(train_green_data)[[2]], name="green")
  red <- layer_input(shape= dim(train_red_data)[[2]], name="red")
  both <- layer_input(shape= dim(train_both_data)[[2]], name="both")

  salida_green <- green %>%
    layer_dense(units=64, activation="relu", name ="dense_green")

  salida_red <- red %>%
    layer_dense(units=64, activation="relu", name="dense_red")

  salida_tipo_I <- layer_concatenate(list(salida_green, salida_red),
                                     name = "Tipo_I")

  tipo_I <- salida_tipo_I %>%
    layer_dense(units = 32, activation = "relu", name="dense_Tipo_I")

  tipo_II <- both %>%
    layer_dense(units=64, activation="relu", name="dense_Tipo_II_1") %>%
    layer_dense(units=32, activation="relu", name = "dense_Tipo_II_2")

  concatenated <- layer_concatenate(list(tipo_I, tipo_II), name="Entrada_Tipo_I_TipoII")

  salida <- concatenated %>%
    layer_dense(units=64, activation = "relu", name="dense_conjunta") %>%
    layer_dense(units=35, activation = "softmax", name="salida")


  model <- keras_model(list(green, red, both), salida)

  model %>%  compile(
    optimizer = "rmsprop",
    loss = "categorical_crossentropy",
    metrics = c("accuracy")
  )
}
```

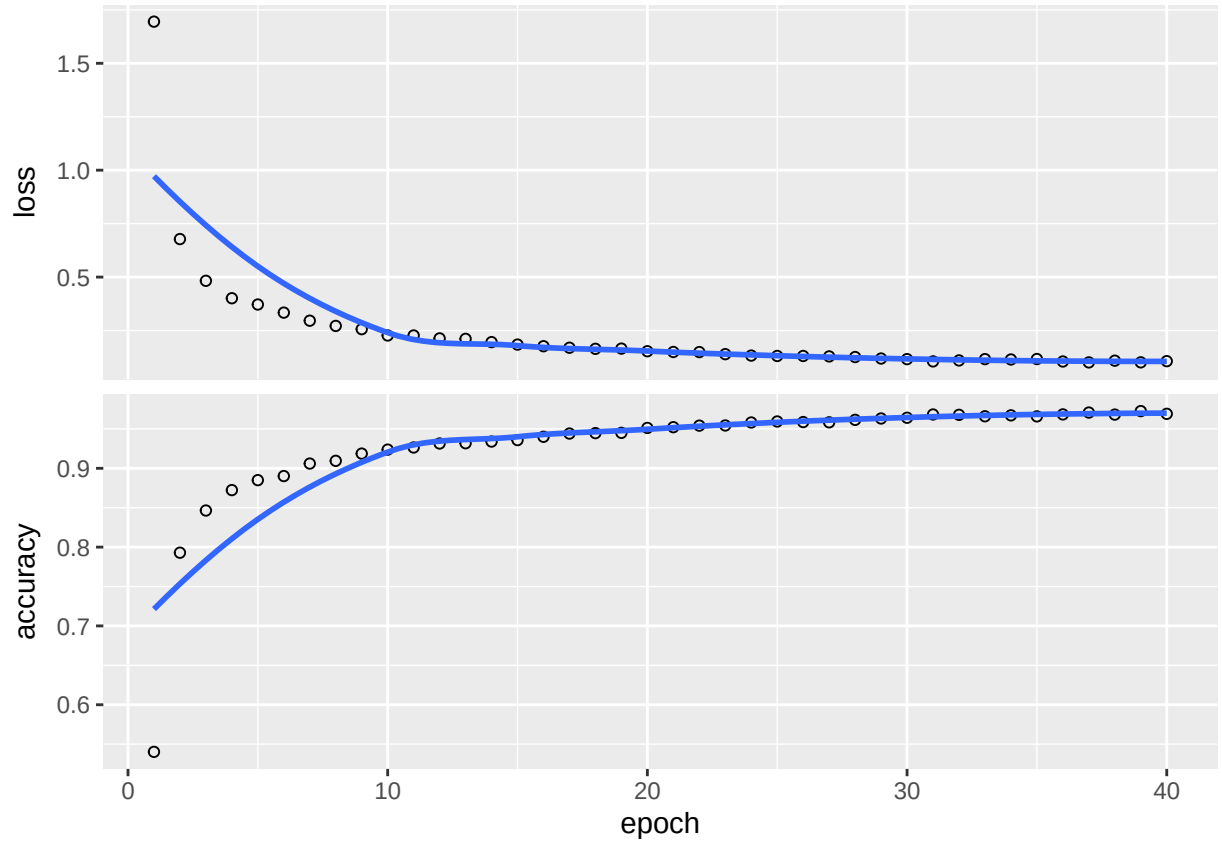
2.3 Entrenamiento del modelo con todas las observaciones

Solo estan las muestras de tumores primarios.

```
modelo <- build_model()

num_epoch = 40
history <- modelo %>% fit(list(train_green_data, train_red_data, train_both_data ),
                          label_c, epoch= num_epoch)

plot(history)
```




```
save_model_hdf5(modelo, "G:/TFM UOC/datos/Clasificador_variabilidad/modelo_10000.h5")
```

3 Predicción con los datos de colonomics

3.1 Carga de los datos

```
betas_colonomics <- read_csv("G:/TFM UOC/datos/colonomics/CLX_methylation_betas_2014Apr25.txt")
```

```
## Rows: 430086 Columns: 248
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): cpg
## dbl (247): E2069_B, K2068_B, P2055_B, P2078_B, S2039_B, W2072_B, Z2038_B, A6...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
betas_colonomics[1:5, 1:5]
```

```
## # A tibble: 5 x 5
##   cpg        E2069_B K2068_B P2055_B P2078_B
```

```
##   <chr>        <dbl>   <dbl>   <dbl>   <dbl>
## 1 cg13869341     888     840     868     800
## 2 cg24669183     855     838     856     823
## 3 cg15560884     817     818     808     804
## 4 cg01014490      69      48      51      40
## 5 cg17505339     903     906     891     898
```

```
dim(betas_colonomics)
```

```
## [1] 430086    248
```

```
betas <- as.matrix(betas_colonomics[, 2:dim(betas_colonomics)[2]])

sondas_c <- betas_colonomics[ , 1]
row.names(betas) <- sondas_c$cpg
betas[1:3, 1:5]
```

```
##            E2069_B K2068_B P2055_B P2078_B S2039_B
## cg13869341     888     840     868     800     829
## cg24669183     855     838     856     823     878
## cg15560884     817     818     808     804     832
```

```
dim(betas)
```

```
## [1] 430086    247
```

3.2 Estimación de datos de las sondas que faltan en colonomics

Primero se crea un data.frame con los nombres de las 1.000 sondas más variables, y con la función `merge` se le añaden los datos betas del proyecto colonomics

```
# se crea el data frame con solo el nombre de las filas: las 1.000 sondas más variables
betas_s <- data.frame(row.names=row.names(sondas_10000))

# con merge se le añaden los valores betas de colonomics
betas_s2 <- merge(betas_s, betas, by.x=0, by.y=0, all.x=TRUE)

# Vemos que existen NAs en las sondas que faltan en colonomics
betas_s2[1:5, 1:5]
```

```
##    Row.names E2069_B K2068_B P2055_B P2078_B
## 1 cg00000165     276     363     288     312
## 2 cg00005847     207     189     164     209
## 3 cg00008629     341     405     628     593
## 4 cg00009553     738     739     821     752
## 5 cg00010853     549     494     516     606
```

```
# Se nombran las filas con la primera columna
row.names(betas_s2) <- betas_s2$Row.names
```

```
# Se elimina la primera columna que tenía el nombre de las sondas
betas_s2 <- betas_s2[ , 2:dim(betas_s2)[2]]

# se reordena el data.frame con las betas de colonomics para que
# las sondas estén en el mismo orden que en el modelo red_neuronal
betas_s <- betas_s2[row.names(sondas_10000), ]
dim(betas_s)
```

```
## [1] 10000   247
```

```
betas_s [1:5, 1:5]
```

```
##            E2069_B K2068_B P2055_B P2078_B S2039_B
## cg20148575      NA      NA      NA      NA      NA
## cg04039555     182     134      70      78      95
## cg23060646      NA      NA      NA      NA      NA
## cg26452868      NA      NA      NA      NA      NA
## cg08632388      NA      NA      NA      NA      NA
```

3.3 Estimación de los valores faltantes con `impute`

Se estiman los valores faltantes con la función `impute.knn` y se dividen por 1.000 para obtener los valores betas en tanto por uno

```
betas_s_sna <- impute.knn(as.matrix(betas_s))$data / 1000
```

```
## Warning in knnimp(x, k, maxmiss = rowmax, maxp = maxp): 410 rows with more than 50 % entries missing
##  mean imputation used for these rows

## Cluster size 9590 broken into 3894 5696
## Cluster size 3894 broken into 1490 2404
## Done cluster 1490
## Cluster size 2404 broken into 943 1461
## Done cluster 943
## Done cluster 1461
## Done cluster 2404
## Done cluster 3894
## Cluster size 5696 broken into 2897 2799
## Cluster size 2897 broken into 1560 1337
## Cluster size 1560 broken into 839 721
## Done cluster 839
## Done cluster 721
## Done cluster 1560
## Done cluster 1337
## Done cluster 2897
## Cluster size 2799 broken into 780 2019
## Done cluster 780
## Cluster size 2019 broken into 997 1022
## Done cluster 997
## Done cluster 1022
## Done cluster 2019
```

```
## Done cluster 2799
## Done cluster 5696
```

```
betas_s_sna[1:5, 1:5]
```

```
##             E2069_B   K2068_B   P2055_B   P2078_B   S2039_B
## cg20148575 0.501268 0.5133133 0.4972271 0.5005211 0.5052311
## cg04039555 0.182000 0.1340000 0.0700000 0.0780000 0.0950000
## cg23060646 0.501268 0.5133133 0.4972271 0.5005211 0.5052311
## cg26452868 0.501268 0.5133133 0.4972271 0.5005211 0.5052311
## cg08632388 0.501268 0.5133133 0.4972271 0.5005211 0.5052311
```

En el mensaje de aviso vemos que había 87 sondas en el subset de sondas más variables que no estaban en los datos de colonomics

3.4 Carga valores de fenotipos de colonomics

```
fenotipos <- read_csv("G:/TFM UOC/datos/colonomics/CLX_ClinicalData.csv")
```

```
## Rows: 250 Columns: 16
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): id_clx, type, id_clx_individual, stage, sex, site, metastasis_site...
## dbl  (6): age, event_free, time_free, event_global, time_global, stromal_score
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# selección de los fenotipos incluidos en la matriz de valores betas de colonomics
fenotipos_s <- intersect(fenotipos$id_clx, colnames(betas_s_sna))
fenotipos_r <- fenotipos[fenotipos$id_clx %in% fenotipos_s, ]

# selección de las muestras de colonomics para las que tenemos valor de fenotipo
# y trasposición de la matriz
betas_s_sna <- betas_s_sna[ , fenotipos_s] %>%  t()

dim(betas_s_sna)
```

```
## [1]   229 10000
```

```
sum(fenotipos_r$id_clx != rownames(betas_s_sna))
```

```
## [1] 0
```

```
table(fenotipos_r$type)
```

```
##
## Mucosa Normal  Tumor
##     37     96     96
```

ya tenemos 229 muestras con su matriz de betas de las 1.000 sondas más variables y sus valores de fenotipos

3.5 Preparación de datos de colonomics

Desglose de las sondas de acuerdo al tipo de sonda

```
dim(betas_s_sna)
```

```
## [1]   229 10000
```

```
colonomics_green <- betas_s_sna[ , tipos_sondas=="Grn"]
colonomics_red <- betas_s_sna[ , tipos_sondas == "Red"]
colonomics_both <- betas_s_sna[ , tipos_sondas =="Both"]
```

3.6 predicción con las betas de colonomics

```
prediccion <- modelo %>% predict(list(colonomics_green, colonomics_red, colonomics_both)) %>%
  k_argmax() %>%
  as.array() %>% as.integer()

l <- as.list(1:34)
names(l) <- levels(label)
f <- names(l)[prediccion]
table(prediccion = f, real_colonomics= fenotipos_r$type)
```

```
##           real_colonomics
## prediccion Mucosa Normal Tumor
##    COAD        12     20    92
##    Control     25     69     2
##    PRAD         0      5     0
##    STAD         0      2     2
```

Bibliografía

Capper, David, David TW Jones, Martin Sill, Volker Hovestadt, Daniel Schrimpf, Dominik Sturm, Christian Koelsche, et al. 2018. «DNA methylation-based classification of central nervous system tumours». *Nature* 555 (7697): 469-74.

Chollet, F, y JJ Allaire. 2018. «Deep Learning with R, Manning Publications».

Kuhn, Max. 2017. *A Short Introduction to the caret Package*. https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf.

Lantz, Brett. 2015. *Machine learning with R*. Packt Publishing Ltd. http://www.packtpub.com/books/content/machine-learning-r.

Maros, Máté E, David Capper, David TW Jones, Volker Hovestadt, Andreas von Deimling, Stefan M Pfister, Axel Benner, Manuela Zucknick, y Martin Sill. 2020. «Machine learning workflows to estimate class probabilities for precision cancer diagnostics on DNA methylation microarray data». *Nature protocols* 15 (2): 479-512.

Price, E Magda, Allison M Cotton, Lucia L Lam, Pau Farré, Eldon Emberly, Carolyn J Brown, Wendy P Robinson, y Michael S Kobor. 2013. «Additional annotation enhances potential for biologically-relevant analysis of the Illumina Infinium HumanMethylation450 BeadChip array». *Epigenetics & chromatin* 6 (1): 1-15.

Zhou, Wanding, Peter W Laird, y Hui Shen. 2017. «Comprehensive characterization, annotation and innovative use of Infinium DNA methylation BeadChip probes». *Nucleic acids research* 45 (4): e22-22.