BIOLOGÍA COMPUTACIONAL CON BIG DATA-OMICS E INGENIERÍA BIOMÉDICA

NOMBRE Y APELLIDOS: **Daniel Redondo Sánchez**

Referencia para este trabajo:

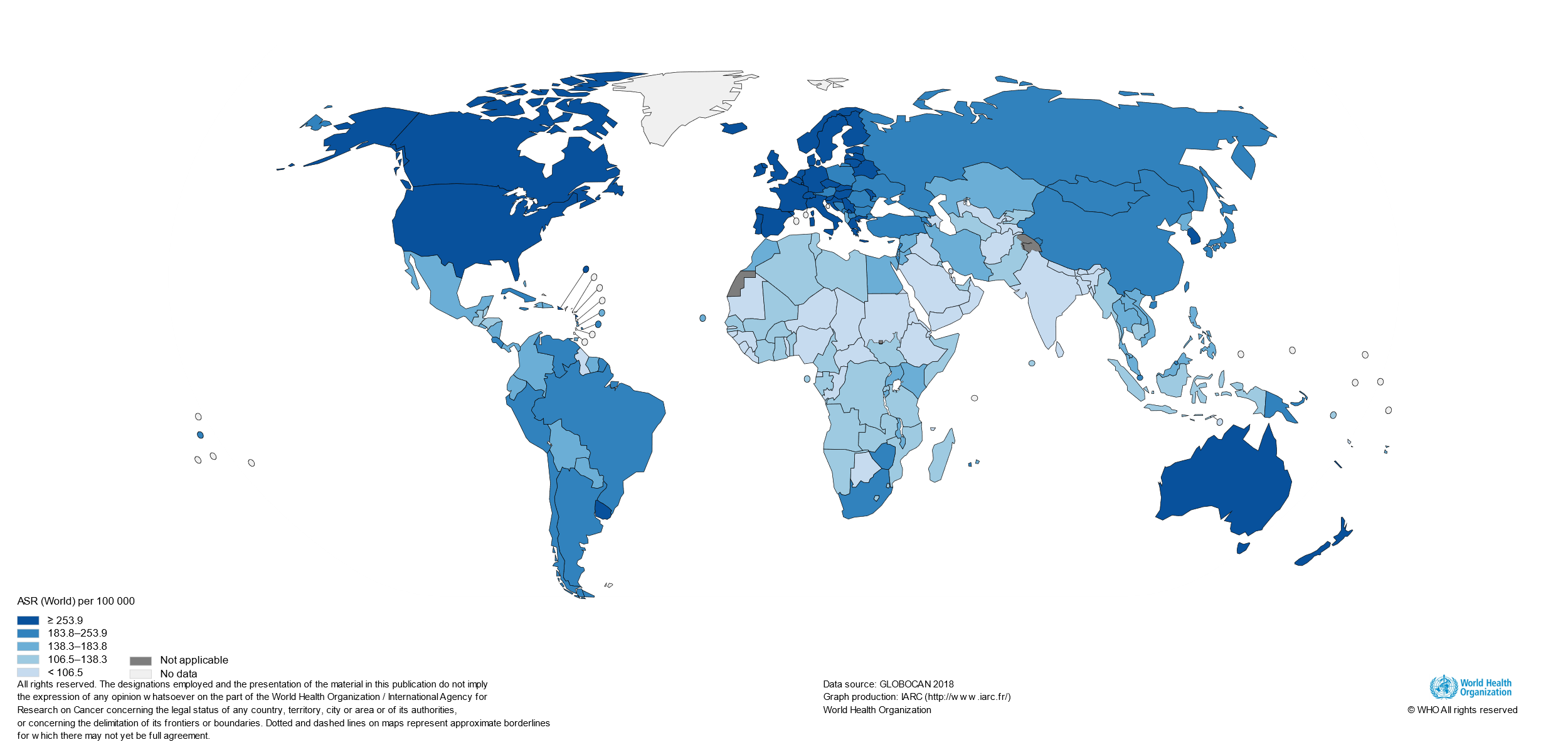
[1] <https://bioconductor.org/packages/devel/bioc/vignettes/KnowSeq/inst/doc/KnowSeq.html>

1.**- Descripción de la enfermedad y de los datos bajados de GDC.**

**Cáncer de tiroides en el mundo**

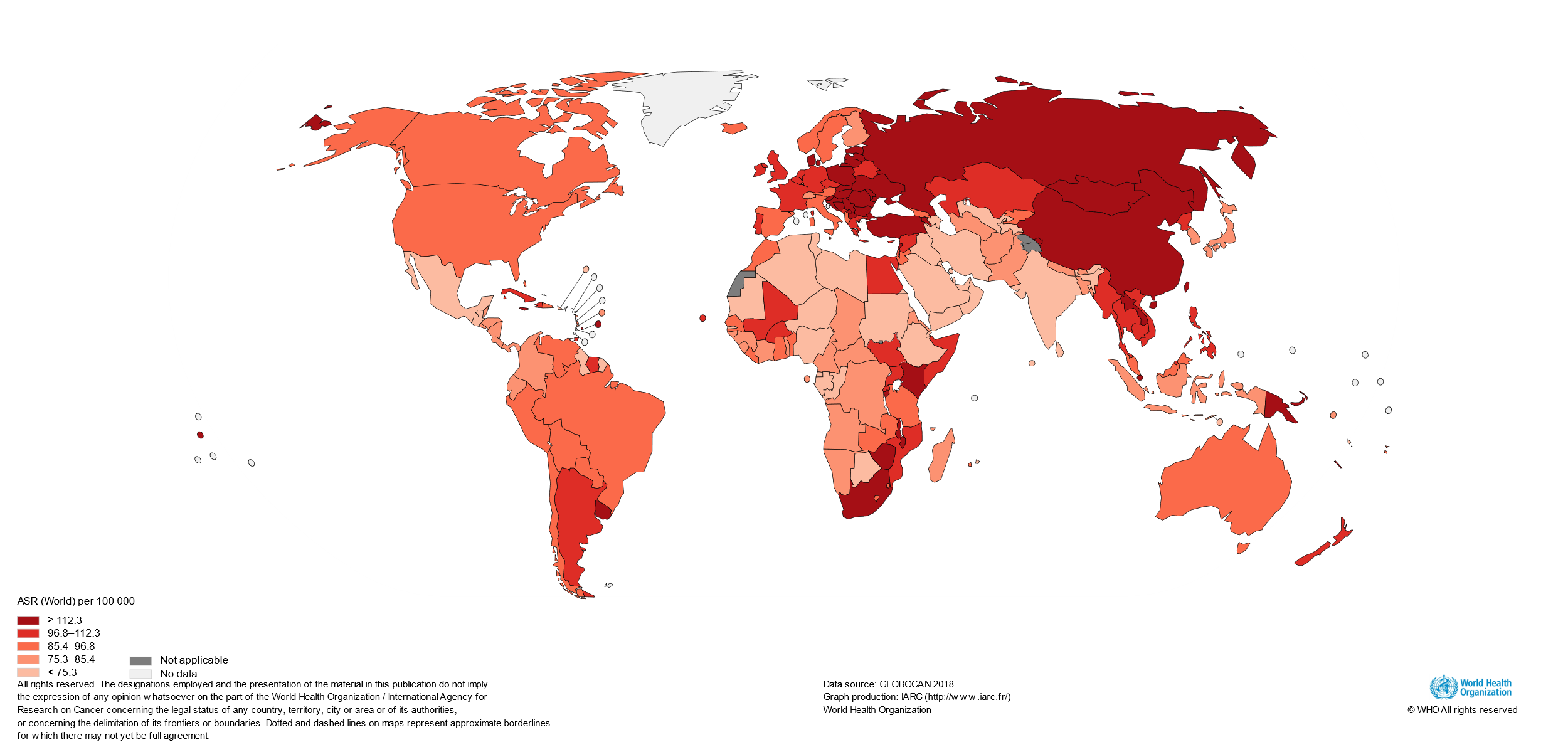
Según datos del *Global Cancer Observatory* [1], se estima que en el año 2018 se diagnosticaron 567.233 casos nuevos de cáncer de tiroides. Es un cáncer con mayor incidencia en mujeres, que representan el 77% del total de casos.

**Figura 1. Mapa de incidencia del cáncer de tiroides (tasa estandarizada por la población mundial por 100.000 habitantes), 2018.**



Durante el año 2018 se produjeron 41.071 defunciones por cáncer de tiroides (62% en mujeres).

**Figura 2. Mapa de mortalidad del cáncer de tiroides (tasa estandarizada por la población mundial por 100.000 habitantes), 2018.**



Las diferencias geográficas existentes entre la incidencia y la mortalidad del cáncer de tiroides se pueden explicar por las desigualdades socioeconómicas existentes, así como por las diferencias evidentes entre los distintos sistemas de salud de todo el mundo.

**Cáncer de tiroides en Europa**

En Europa se estiman 78.418 casos nuevos de cáncer de tiroides durante el año 2018, según datos del European Cancer Information System (ECIS) [2], con una tasa estandarizada por la población europea (ASR-E) de 9,3 casos nuevos por 100.000 habitantes. Las defunciones por cáncer de tiroides fueron 6.988, con una ASR-E de 0,6 defunciones por 100.000 habitantes.

**Cáncer de tiroides en España**

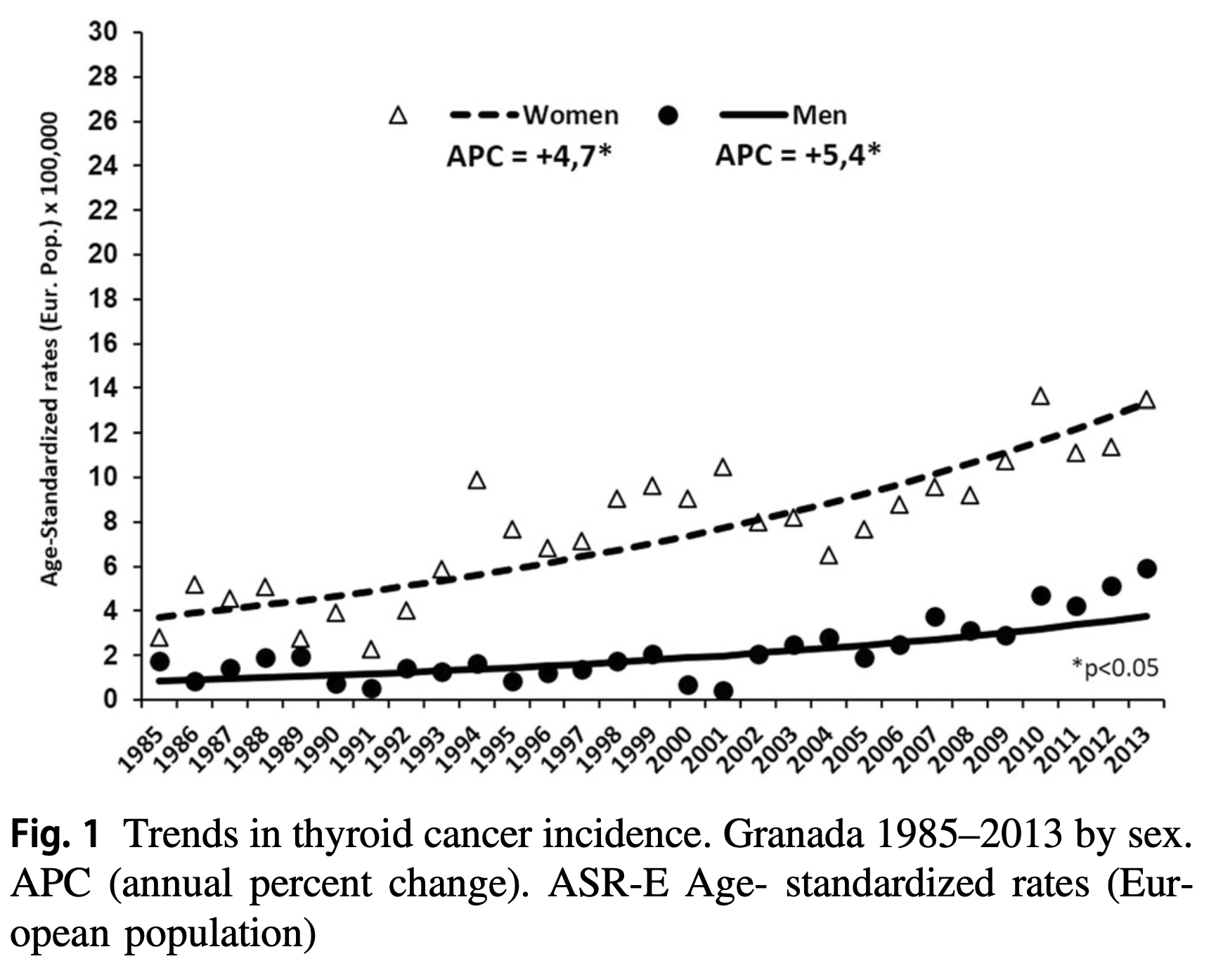
Según las últimas estimaciones de incidencia de cáncer para España realizadas por la Red Española de Registros de Cáncer (REDECAN) para el año 2020, se diagnostican 5.304 casos nuevos de cáncer de tiroides [3]. Los últimos datos de mortalidad los proporciona el Ministerio de Sanidad, Consumo y Bienestar Social, y corresponden al año 2017, cuando se produjeron 332 defunciones por cáncer de tiroides (204 en mujeres).

**Cáncer de tiroides en Granada**

La incidencia de cáncer de tiroides en la provincia de Granada es registrada por el Registro de Cáncer de Granada, donde trabajo desde el año 2016. El cáncer de tiroides es el sexto cáncer más frecuente en mujeres, con 214 casos en el periodo 2012-2014, y una ASR-E de 14,4 casos nuevos por 100.000 mujeres [4].

En un estudio específico de cáncer de tiroides [5], analizamos las tendencias de la incidencia en Granada durante el periodo 1985-2013, encontrando un aumento estadísticamente significativo para ambos sexos (Porcentaje de Cambio Anual [PCA]: +5,4% en hombres, +4,7% en mujeres), principalmente debido al aumento en el carcinoma papilar. Sin embargo, la mortalidad descendió ligeramente tanto en hombres (PCA = -0,3%) como en mujeres (PCA = -2,3%).

**Figura 3. Tendencias en la incidencia de cáncer de tiroides. Tasa estandarizada por la población europea en Granada, por sexos, durante los años 1985-2013.**



**Supervivencia del cáncer de tiroides**

El cáncer de tiroides es uno de los cánceres con mayor supervivencia en Europa (86,5% a los 5 años [6]) y el cáncer con mejor supervivencia en España (93,1% a los 5 años para mujeres, 86,1% para hombres [7]). Es por tanto un cáncer poco agresivo y con buen pronóstico en general.

**Referencias:**

[1] – Global Cancer Observatory, International Agency for Research on Cancer, World Health Organization: <http://gco.iarc.fr/>

[2] – European Cancer Information System, European Commission: <https://ecis.jrc.ec.europa.eu/>

[3] – Estimaciones de la incidencia del cáncer en España, 2020. Red Española de Registros de Cáncer (REDECAN), 2020. Disponible online en: <https://funca.cat/redecan/redecan.org/es/Informe_incidencia_REDECAN_2020.pdf>

[4] – Informe de Incidencia de Cáncer en la Provincia de Granada, 2012-2014. Rodríguez-Barranco M, López-López D, **Redondo-Sánchez D**, Sánchez MJ.

Disponible online en: <http://cancergranada.org/es/estadisticas_incidencia.cfm>

[5] - Thyroid Cancer Epidemiology in South Spain: a population-based time trend study. Elena Salamanca-Fernández, Miguel Rodriguez-Barranco, Yoe-Ling Chang-Chan, **Daniel Redondo-Sánchez**, Santiago Domínguez-López, Eloísa Bayo, Dariusz Narankiewicz, José Expósito & María José Sánchez. *Endocrine*. DOI: 10.1007/s12020-018-1681-6. Disponible online en: <https://link.springer.com/article/10.1007%2Fs12020-018-1681-6>

[6] - Cancer survival in Europe 1999–2007 by country and age: results of EUROCARE-5 - a population-based study, De Angelis et al

[7] - Supervivencia de Cáncer en España, 2002-2013, Marcela Guevara et al.

**Datos descargados de GDC**

Se han descargado 750 registros de GDC, con la siguiente distribución:

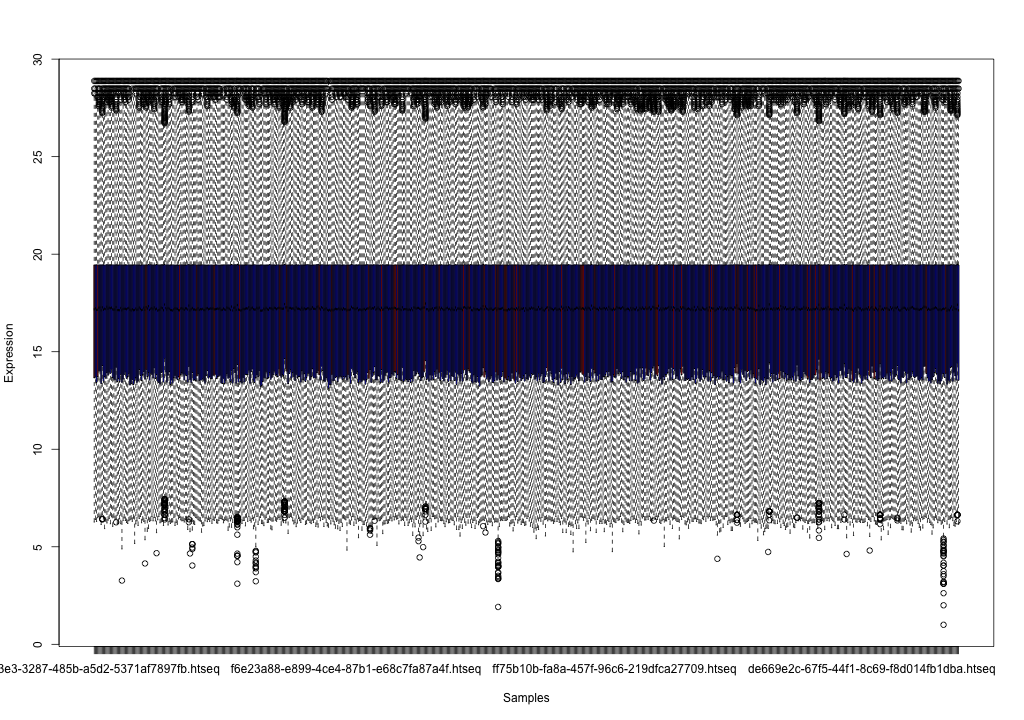
**Tabla 1. Distribución del número de casos de los datos descargados de GDC.**

|  |  |  |
| --- | --- | --- |
|  | Primary Tumor | Solid Tissue Normal |
| Número de casos | 688 | 62 |

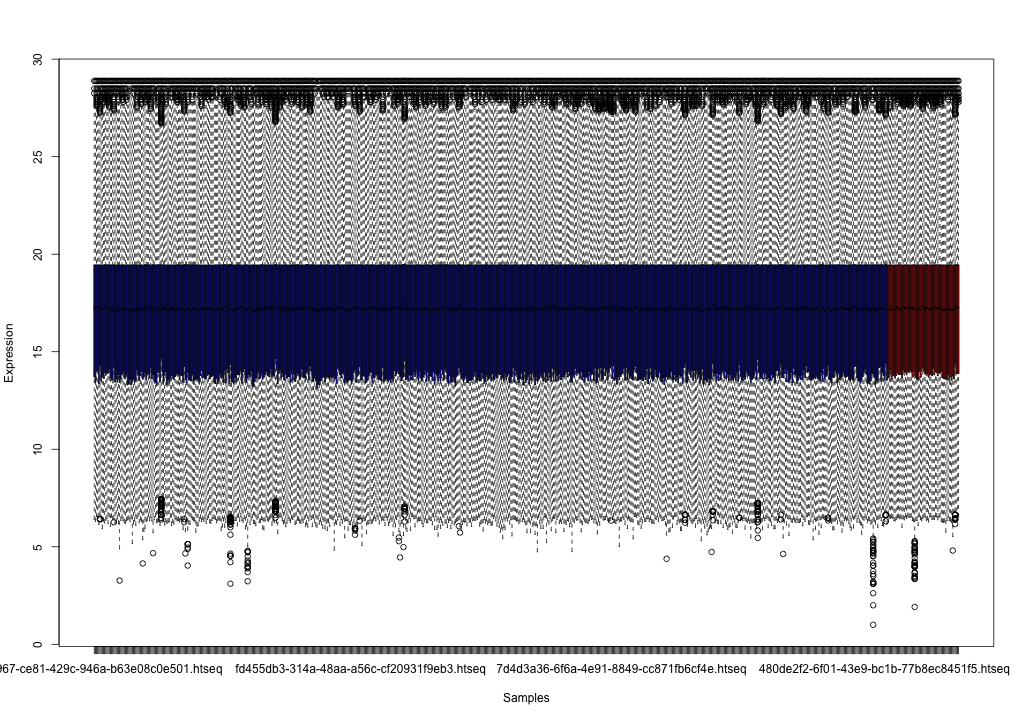
2.**- Boxplot de los datos:**

Se realizan a continuación dos gráficos de cajas y bigotes con los datos: uno con dataPlot(mode = "boxplot") y otro con dataPlot(mode = "orderedBoxplot"). En azul se ven los casos que constituyen un tumor primario, y en rojo los casos con tejido sólido normal.

**Figura 4. Boxplot de la expresión de genes de cada muestra (azul = Primary Tumor, rojo = Solid Tissue Normal).**



**Figura 5. Boxplot ordenado de la expresión de genes de cada muestra (azul = Primary Tumor, rojo = Solid Tissue Normal).**



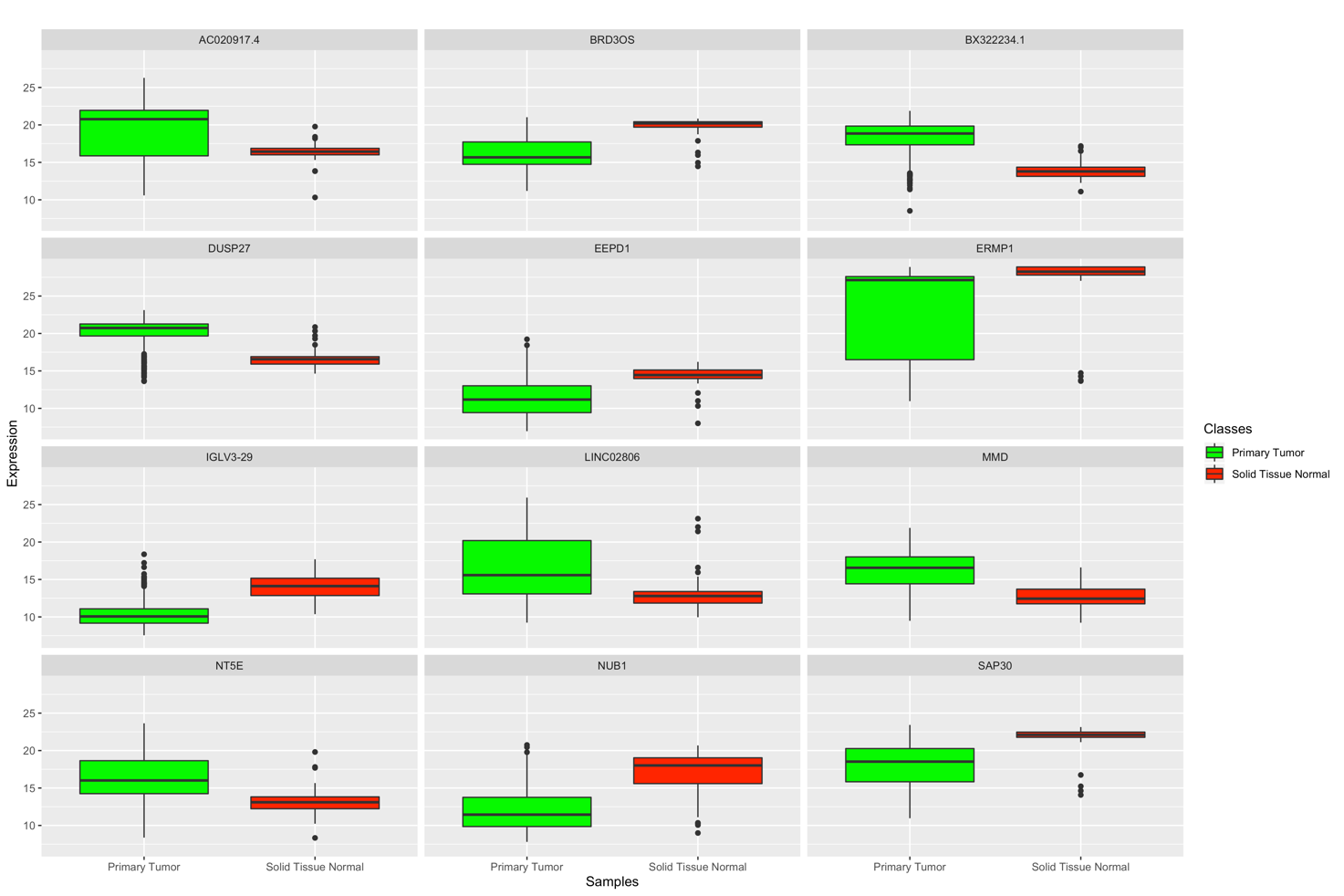
3.**- Differential Expressed Genes extraction and visualization:**

Se controla por el efecto batch con la función batchEffectRemoval con el parámetro method = "sva", que utiliza *surrogate variable analysis*.

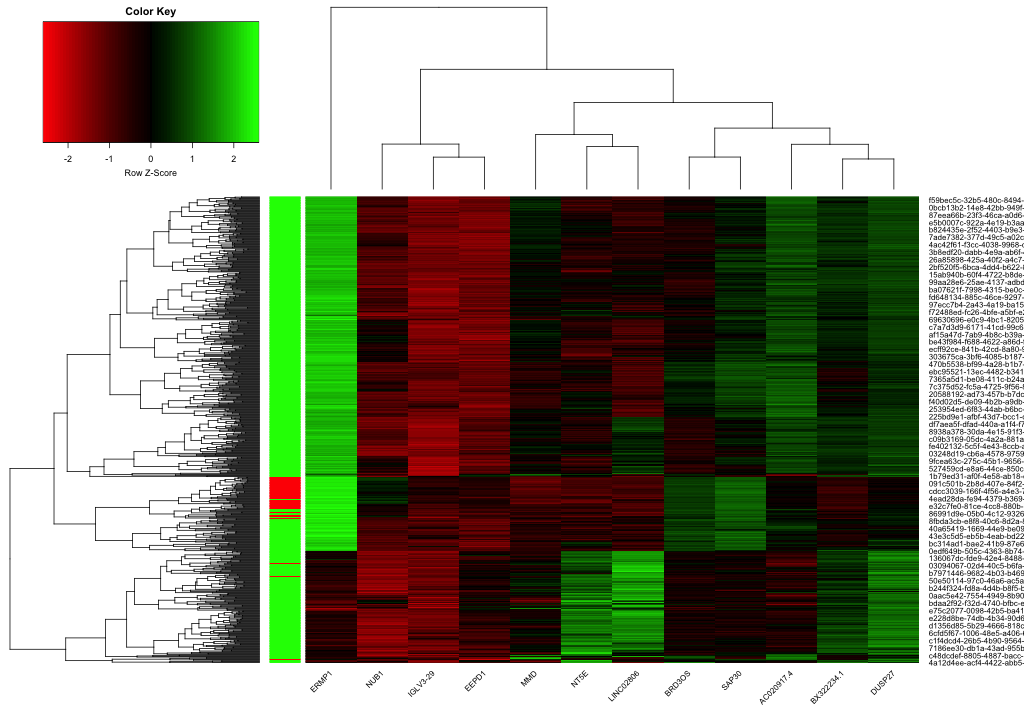
Tras fijar el número de genes a extraer (sin límite, gracias al parámetro number=Inf de la función limmaDEGsExtraction), y un p-valor de 0.01, se han extraído 300 genes que se consideran relevantes con respecto al cáncer de tiroides.

Para la siguiente representación, se utilizan los 12 primeros genes tras la extracción de DEGs. Se visualizan boxplots y mapas de calor.

**Figura 6. Boxplot de la expresión de genes de los 12 primeros genes (verde = Primary Tumor, rojo = Solid Tissue Normal).**



**Figura 7. Mapa de calor de la expresión de genes de los 12 primeros genes.**



4.**- Identificación de biomarcadores (validación cruzada y comparar selección por mrmr, da y rf, en cto entrenamiento):**

**Creación de conjuntos de entrenamiento y test**

Se crea un conjunto de entrenamiento y otro de test, con una proporción de 80%-20%, y manteniendo el equilibrio entre clases. Se utiliza para ello la función caret::createDataPartition, añadiendo la clase como parámetro para garantizar el correcto balanceo de clases.

> set.seed(1991)

> indices <- createDataPartition(SamplesDataFrame$Class, p = .8, list = FALSE)

> particion <- list(training= DEGsMatrixML[indices, ], test= DEGsMatrixML[-indices,])

>

> # Conjuntos

> particion.entrenamiento <- particion$training

> particion.test <- particion$test

>

> # Etiquetas

> labels\_train <- SamplesDataFrame$Class[indices]

> labels\_test <- SamplesDataFrame$Class[-indices]

Se selecciona entonces el siguiente número de casos de cada tipo en cada partición.

**Tabla 2. Número de casos y frecuencia relativa del reparto de casos en entrenamiento y test.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Entrenamiento** | **Test** | **Total** |
|  | N (%) | N (%) | N (%) |
| Primary Tumor | 551 (91,7%) | 137 (91,9%) | 688 (91,7%) |
| Solid Tissue Normal | 50 (8,3%) | 12 (8,1%) | 62 (8,3%) |
| **Total** | **601 (100%)** | **149 (100%)** | **750 (100%)** |

**Selección de características**

Se emplean varios métodos de selección de características: *mRMR* (mínima redundancia, máxima relevancia), *rf* (basado en random forest) y *da* (disease association, basado en scores obtenidos en la literatura).

# Método mRMR (mínima redundancia, máxima relevancia)

mrmrRanking <- featureSelection(particion.entrenamiento, labels\_train, colnames(particion.entrenamiento), mode = "mrmr")

# Método random forest

rfRanking <- featureSelection(particion.entrenamiento, labels\_train, colnames(particion.entrenamiento), mode = "rf")

# Método Disease association ranking (en base a scores obtenidos en la literatura)

daRanking <- featureSelection(particion.entrenamiento, labels\_train, colnames(particion.entrenamiento), mode = "da", disease = "thyroid cancer")

Por ejemplo, los 10 genes más importantes para el método *mRMR* son:

> names(mrmrRanking)[1:10]

[1] "AC027682.4" "PANK2" "NT5C3A" "TFF3" "OR4D10" "C1orf112" "TRMT1L" "AQP8" "NR5A1" "IGLV3-29"

Para el método *rf*:

> rfRanking[1:10]

[1] "AC027682.4" "MRPL43" "MZF1" "PANK2" "NUB1" "NT5C3A" "EEPD1" "ZNF671" "NPHP1" "SNTN"

Y para el método *da*:

> names(daRanking)[1:10]

[1] "TFF3" "NT5E" "NUB1" "ERMP1" "SAP30" "MMD" "DUSP27" "BRD3OS" "EEPD1" "LINC02761"

**Validación cruzada**

Se realiza validación cruzada con 5-fold, usando la función *svm\_CV* del paquete *KnowSeq*, actualizada mediante [un pull-request en GitHub](https://github.com/CasedUgr/KnowSeq/pull/11) (**Anexo 1**) para que imprima por pantalla los valores optimizados de coste y gamma, y los devuelva también como output.

numero\_folds <- 5

# mRMR

results\_cv\_svm\_mrmr <- svm\_CV(particion.entrenamiento, labels\_train, names(mrmrRanking), numFold = numero\_folds)

# random forest

results\_cv\_svm\_rf <- svm\_CV(particion.entrenamiento, labels\_train, rfRanking, numFold = numero\_folds)

# disease association

results\_cv\_svm\_da <- svm\_CV(particion.entrenamiento, labels\_train, names(daRanking), numFold = numero\_folds)

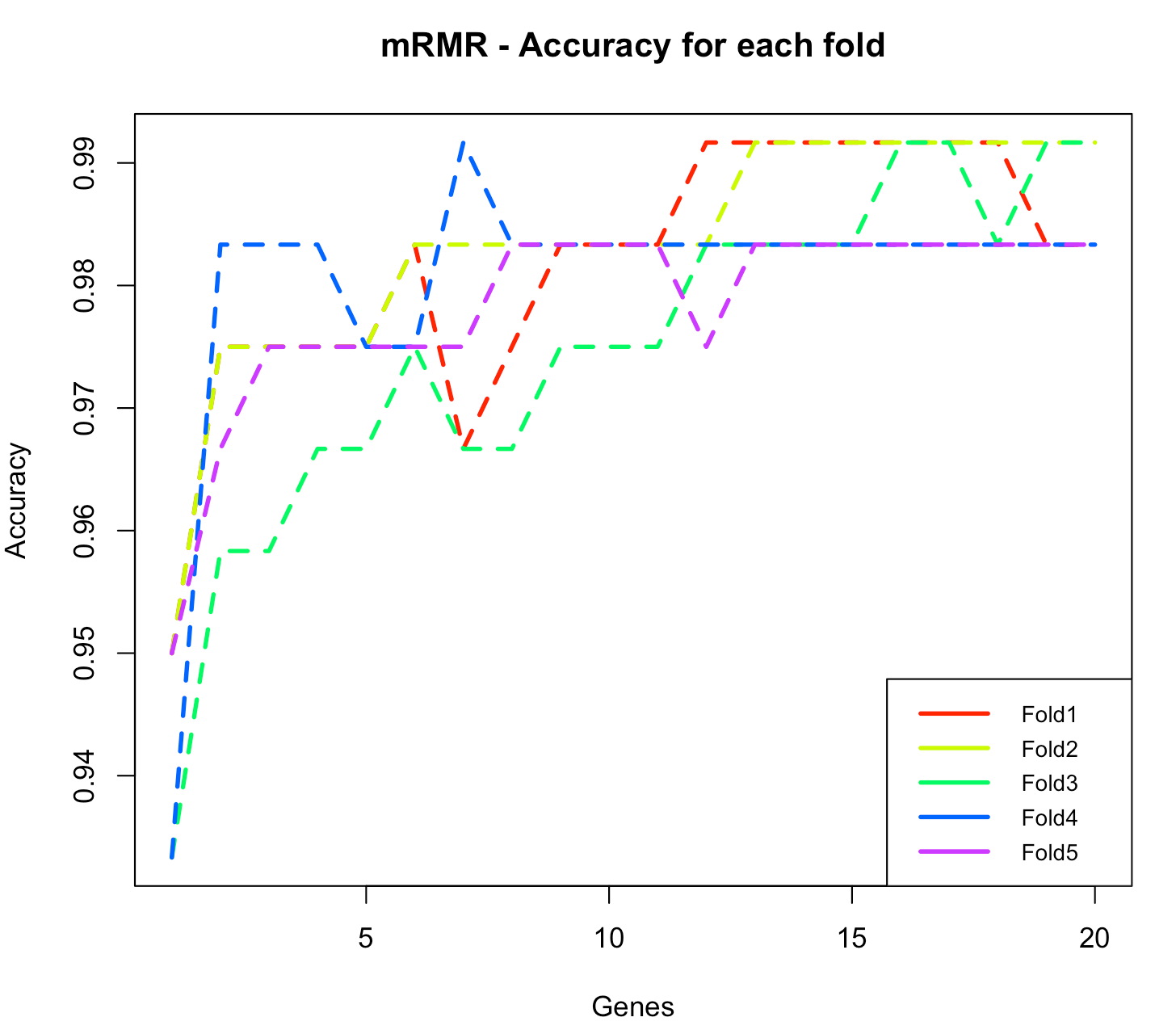
Tras realizar el *tuning* de los parámetros, los parámetros óptimos de coste y gamma del algoritmo SVM para cada método de selección de características se presentan en la Tabla 3.

**Tabla 3. Parámetros óptimos encontrados para cada método de selección de características usando el algoritmo SVM en el conjunto de entrenamiento con validación cruzada 5-fold.**

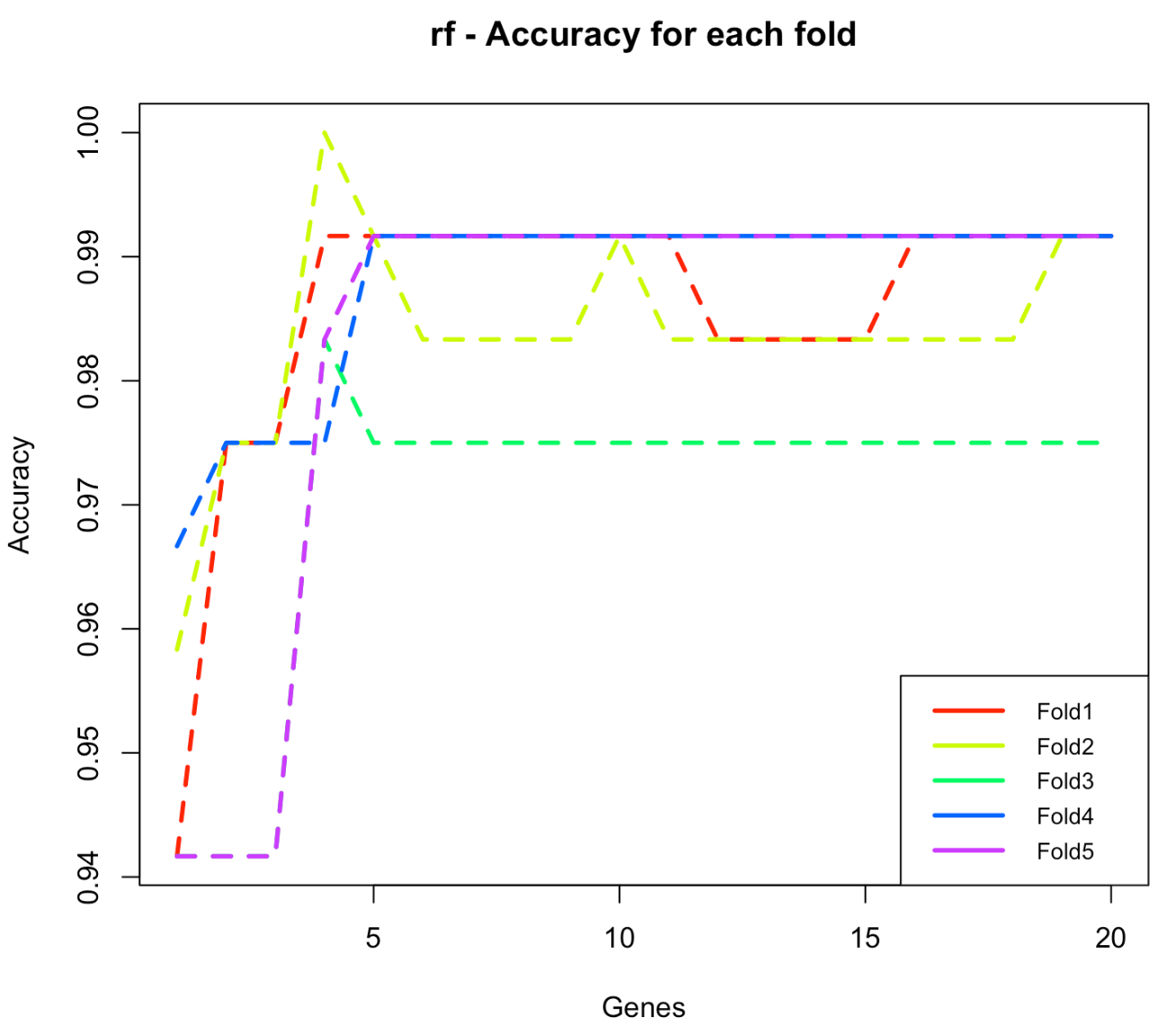
|  |  |  |
| --- | --- | --- |
|  | Coste (c) | Gamma (g) |
| mRMR | 1.5 | 0.02 |
| rf | 1.5 | 0.02 |
| da | 2 | 0.025 |

Se representa a continuación la precisión obtenida en cada fold en función del método de selección de características y el número de genes seleccionado. La función *dataPlot* se actualizó (**Anexo 2**) para mostrar mejor la precisión de cada fold cuando hay superposición de líneas.

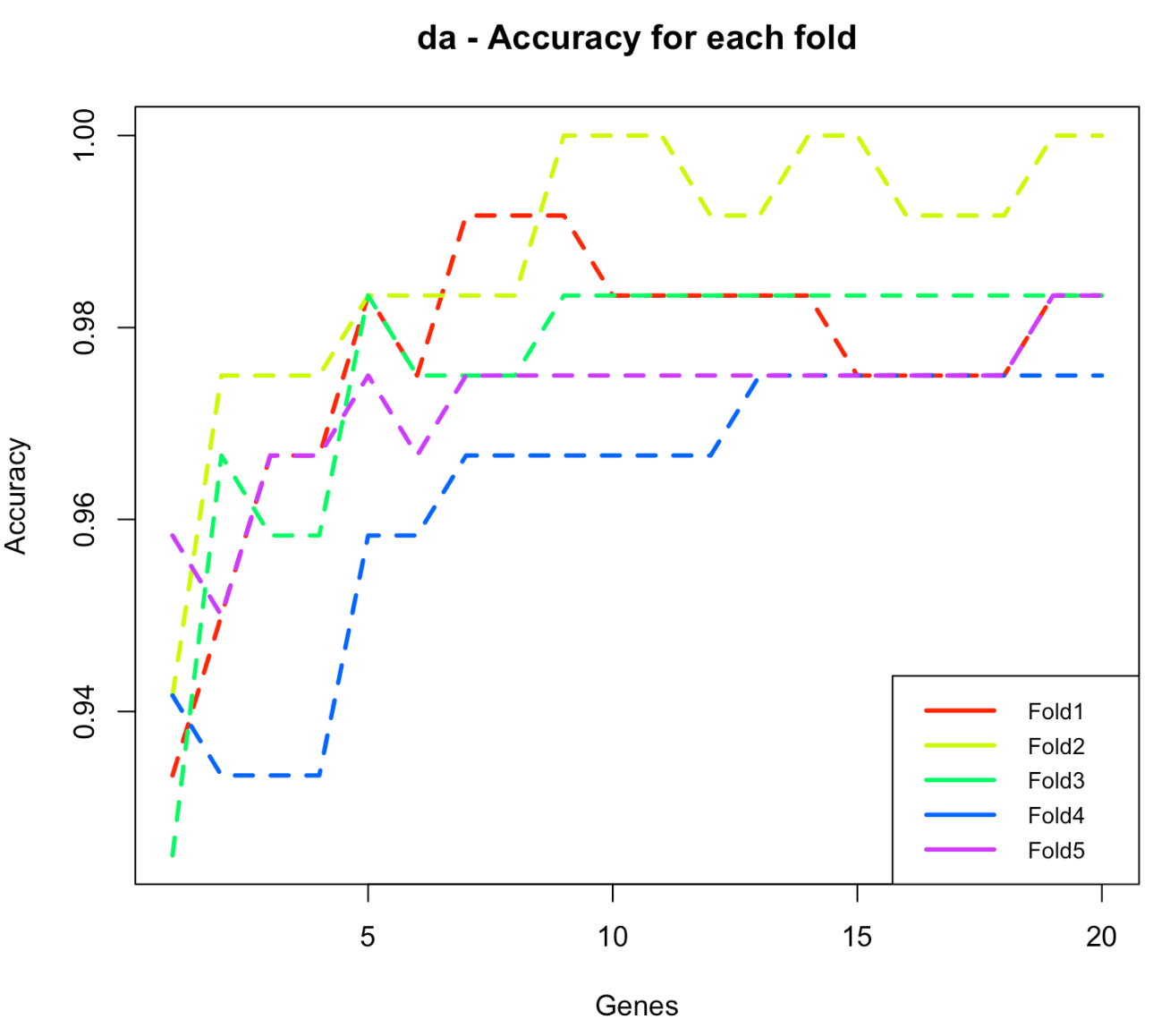
**Figura 7. Precisión de cada fold en el conjunto de entrenamiento usando el algoritmo SVM con los 20 mejores genes seleccionados por la técnica de mRMR.**



**Figura 8. Precisión de cada fold en el conjunto de entrenamiento usando el algoritmo SVM con los 20 mejores genes seleccionados por la técnica de random forest.**



**Figura 7. Precisión de cada fold en el conjunto de entrenamiento usando el algoritmo SVM con los 20 mejores genes seleccionados por la técnica de da.**



Nota: *da* elimina 48 genes que no están relacionados con el cáncer de tiroides en la literatura.

**Tabla 4. Precisión media obtenida en el conjunto de entrenamiento usando el algoritmo SVM con validación cruzada 5-fold, según número de genes seleccionado y técnica de selección de características. Marcado en negrita el mejor método de selección de características para cada número de genes.**

|  |  |  |  |
| --- | --- | --- | --- |
| Número de genes | *mRMR* | *rf* | *da* |
| 1 | 94.333% | **95%** | 94% |
| 5 | 97.333% | **98.833%** | 97.667% |
| 10 | 98.167% | **98.833%** | 98.167% |
| 15 | **98.667%** | 98.5% | 98.167% |
| 25 | 98.5% | **98.833%** | 98.333% |
| 50 | 98.5% | **98.667%** | 98.333% |
| 100 | 98.5% | **98.667%** | 98.5% |

Se puede concluir que:

* La precisión final de los modelos está en torno al 98-99%.
* Con menos de 15 genes ya se obtiene (en general) la mejor precisión que puede aportar cada modelo.
* El mejor algoritmo de selección de características para este problema es *rf*. Aunque *mRMR* obtenga mejor precisión que *rf* seleccionando 15 genes, en realidad el método *rf* con 5 genes obtiene una precisión aún mayor (98.833 vs. 98.667).
* El clasificador definitivo se hará entonces usando selección de características con *rf* y usando 5 genes.

5.**- Heatmap, clasificador definitivo, y resultado en conjunto de test (matriz de confusión):**

Los modelos de SVM entrenados con el conjunto de entrenamiento se evalúan en el conjunto de test.

# mRMR

results\_svm\_mrmr <- svm\_test(train = particion.entrenamiento, labels\_train,

test = particion.test, labels\_test, names(mrmrRanking))

# random forest

results\_svm\_rf <- svm\_test(train = particion.entrenamiento, labels\_train,

test = particion.test, labels\_test, rfRanking)

# disease association

results\_svm\_da <- svm\_test(train = particion.entrenamiento, labels\_train,

test = particion.test, labels\_test, names(daRanking))

Se generan las matrices de confusión de cada modelo considerando un solo gen, y se representan gráficamente:

# Matriz de confusión

results\_svm\_mrmr$cfMats[[1]]$table

results\_svm\_rf$cfMats[[1]]$table

results\_svm\_da$cfMats[[1]]$table

# Gráficamente

dataPlot(results\_svm\_mrmr$cfMats[[1]]$table, labels\_test,

mode = "confusionMatrix")

dataPlot(results\_svm\_rf$cfMats[[1]]$table, labels\_test,

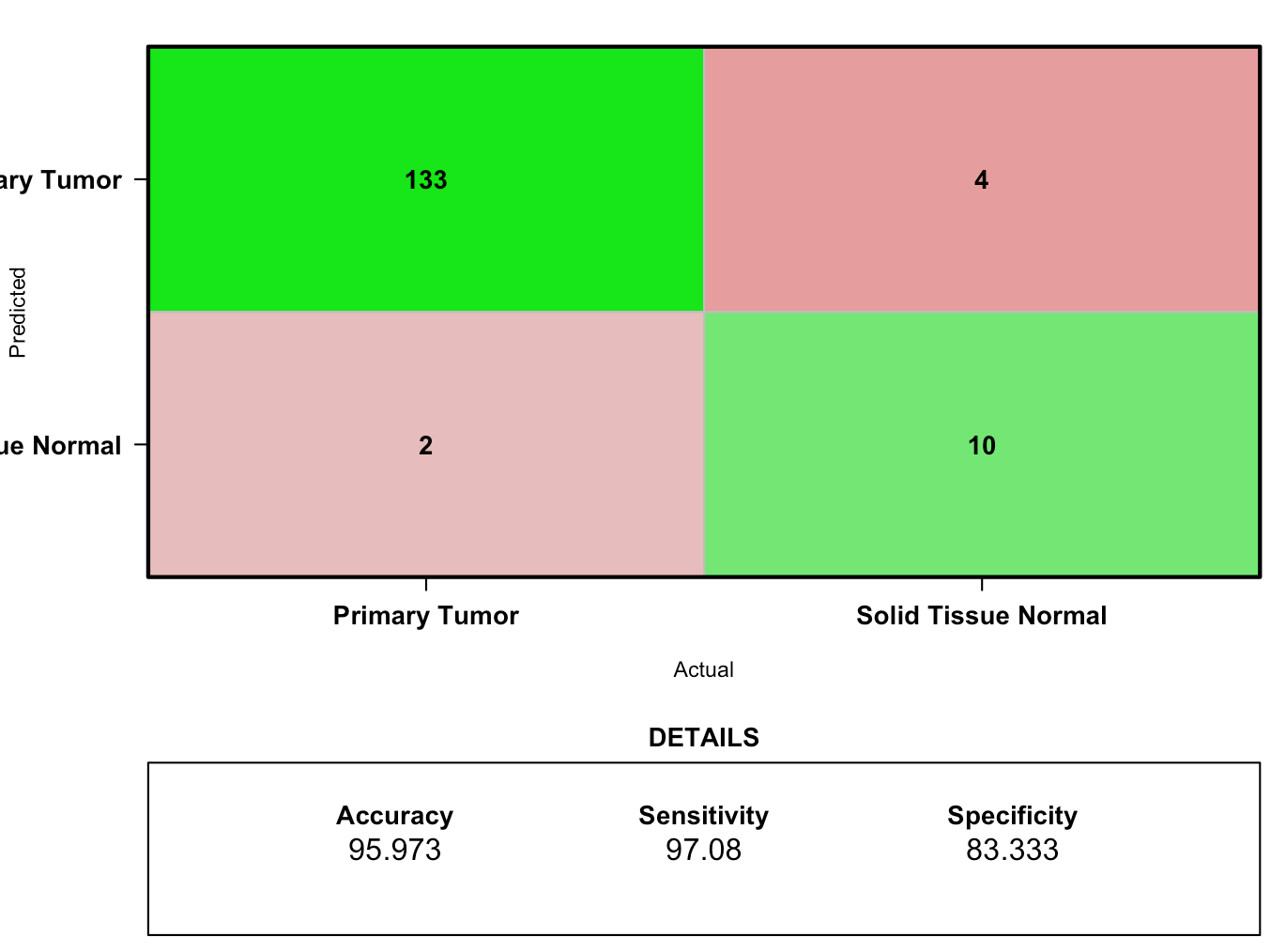
mode = "confusionMatrix")

dataPlot(results\_svm\_da$cfMats[[1]]$table, labels\_test,

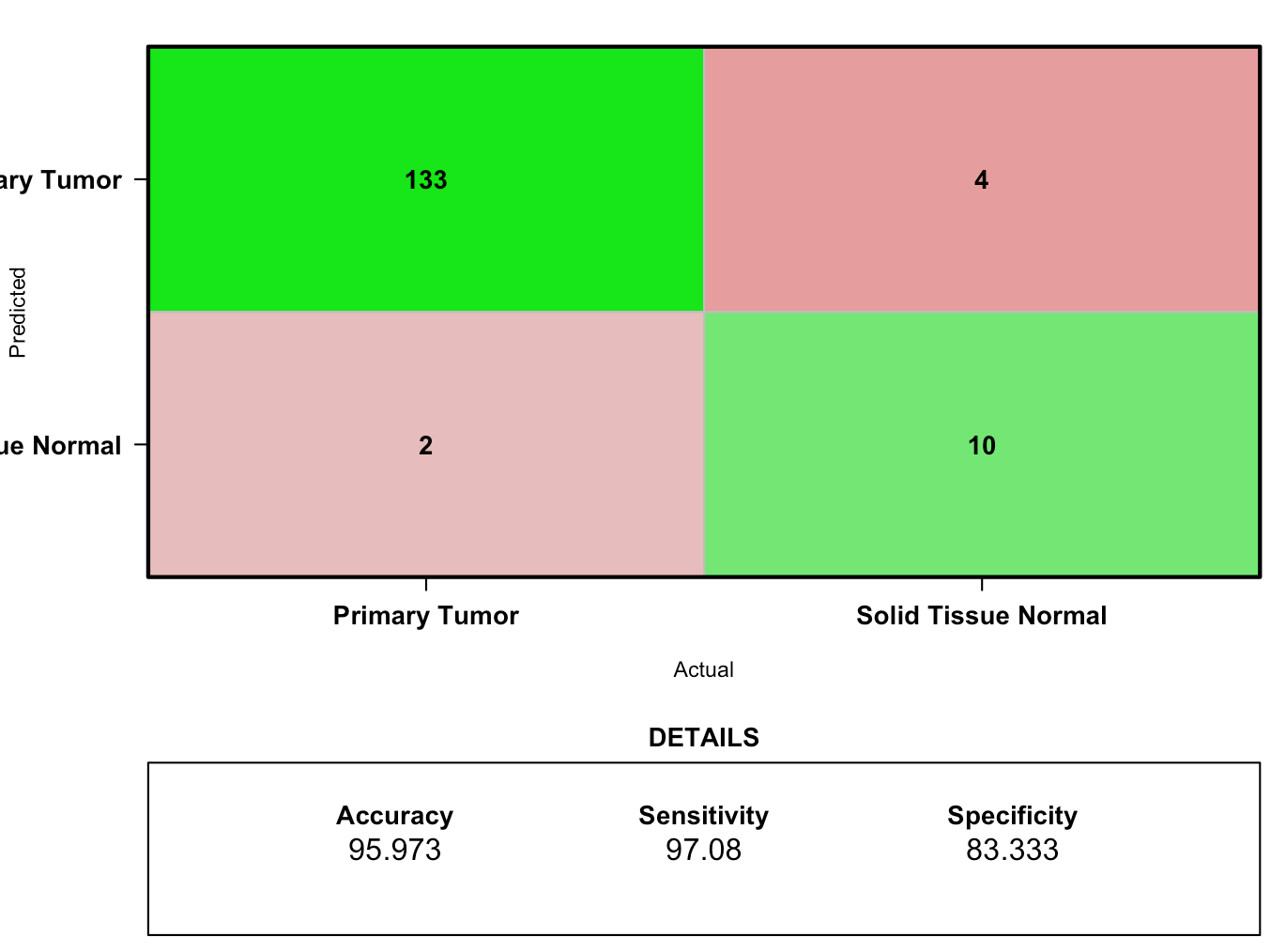
mode = "confusionMatrix")

**Figura 7. Matriz de confusión, precisión, sensibilidad y especificidad del conjunto de test al usar el clasificador SVM con el mejor gen encontrado usando *mRMR*.**

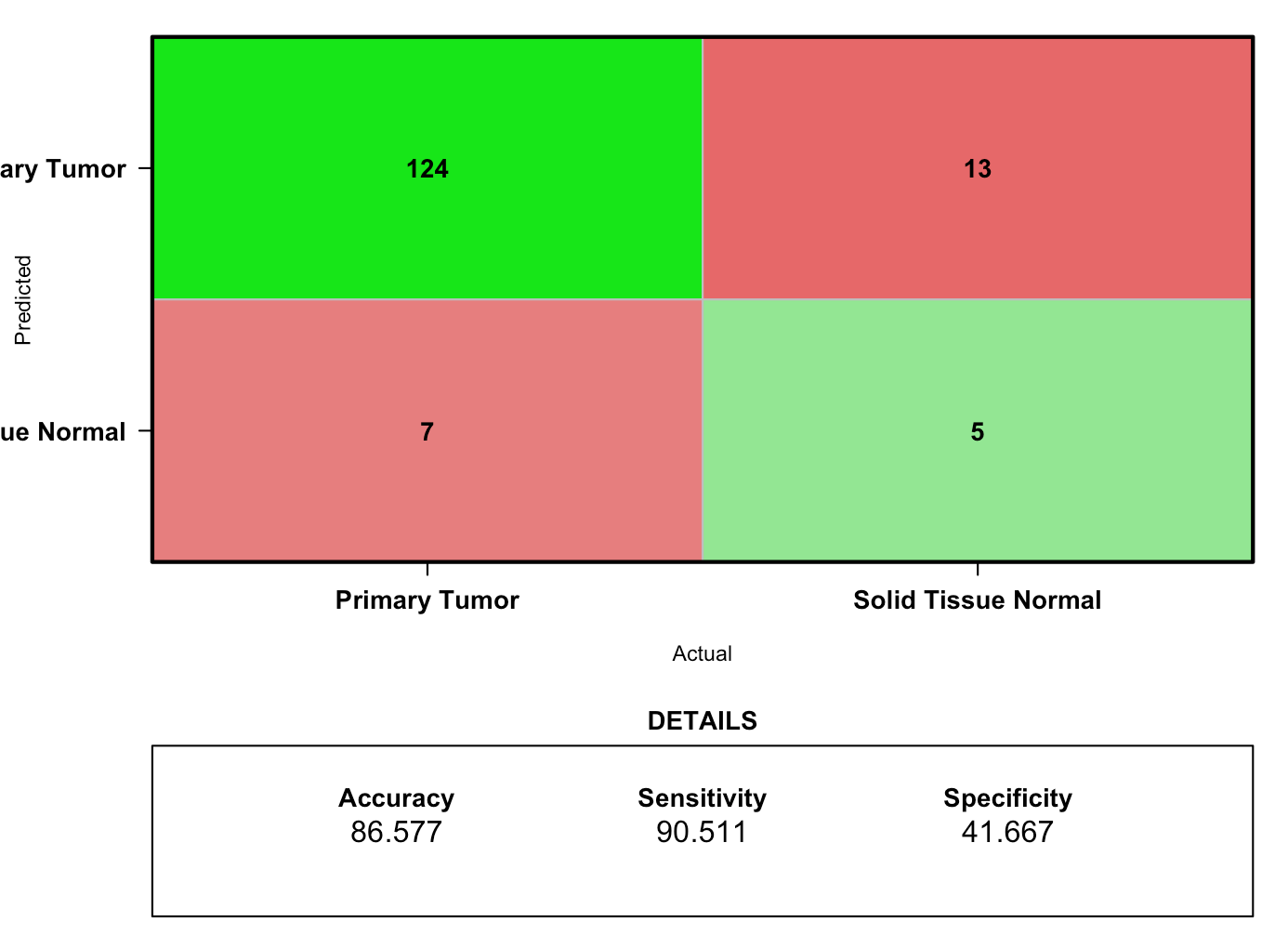
*mRMR*



**Figura 8. Matriz de confusión, precisión, sensibilidad y especificidad del conjunto de test al usar el clasificador SVM con el mejor gen encontrado usando *rf*.**



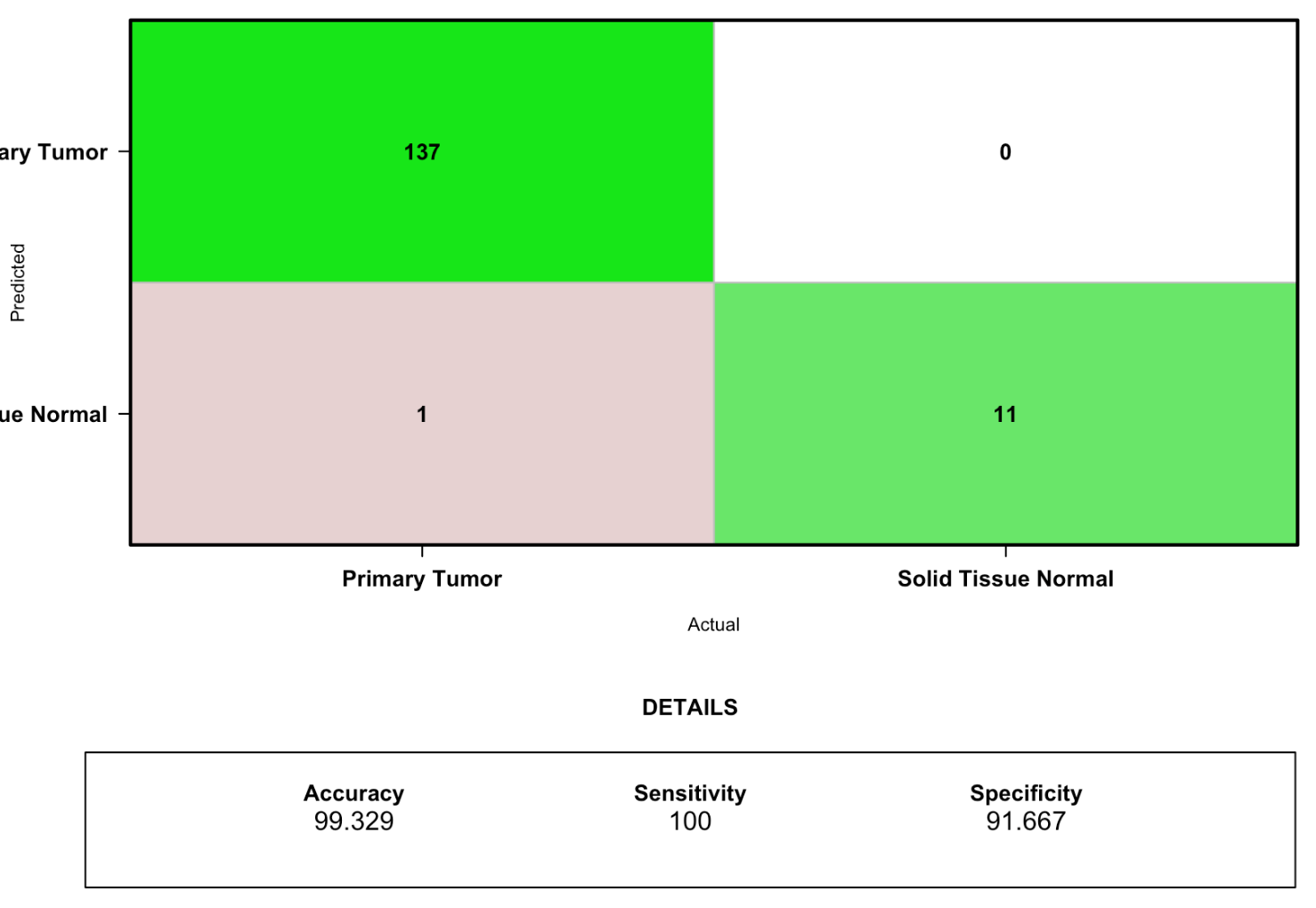
**Figura 9. Matriz de confusión, precisión, sensibilidad y especificidad del conjunto de test al usar el clasificador SVM con el mejor gen encontrado usando *da*.**



Se obtienen los mismos resultados con *mRMR* y *rf* (con una precisión alrededor de 96%) , resultados mejores que los que proporciona *da* (precisión de 86.6%). Con *mRMR* o *rf*, habría 4 falsos positivos (diagnosticados como tumor, siendo tejidos normales) y 2 falsos negativos (diagnosticados como tejidos normales siendo realmente tumores).

Seleccionando el mejor modelo encontrado en validación cruzada (5 genes seleccionados mediante *rf*), se obtienen los siguientes resultados.

**Figura 10. Matriz de confusión, precisión, sensibilidad y especificidad del conjunto de test al usar el clasificador SVM con el mejor gen encontrado usando *rf*.**



Este algoritmo no consigue clasificar perfectamente el conjunto de test, pero sólo se equivoca en 1 caso, clasificado como tejido normal cuando en realidad era tumor primario (falso negativo).

6.**- DEGs enrichment methodology (gene ontology, pathway visualization, related diseases):**

**Enfermedades relacionadas**

Se recuperan las 20 enfermedades más importantes vinculadas a los 5 genes más relevantes del método *rf*, que ha sido el mejor método encontrado tras la validación cruzada.

diseases <- DEGsToDiseases(rfRanking[1:5], getEvidences = TRUE, size = 20)

diseases

# Extracción de todas las enfermedades relacionadas con los 5 genes

enfermedades <- c()

for(i in 1:5){

# Se extraen las enfermedades relacionadas

enfermedades <- c(enfermedades, diseases[[i]]$summary[, 1])

# Se eliminan duplicados

enfermedades <- unique(enfermedades)

}

enfermedades

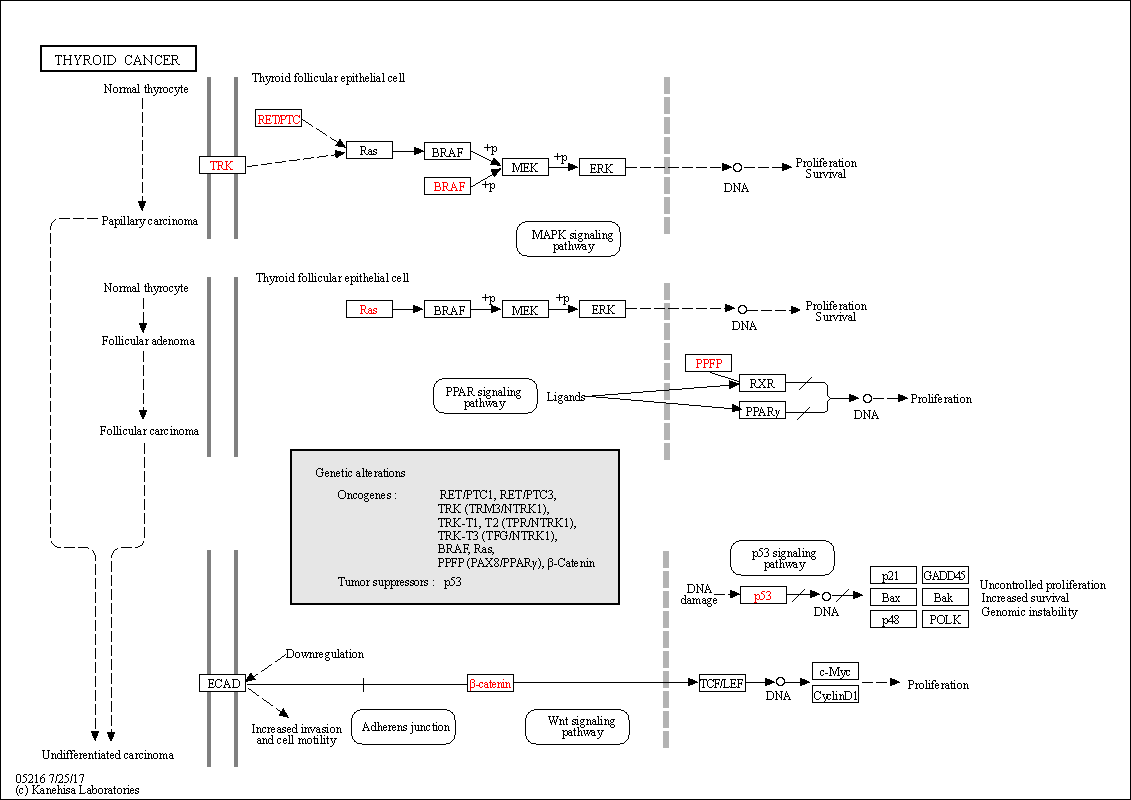
**Tabla 5. Enfermedades relacionadas con los 5 genes más relevantes usando el método de selección de características *rf*. Enfermedades duplicadas se han eliminado para evitar redundancia. Enfermedades relacionadas con el cáncer de tiroides han sido marcadas en negrita.**

|  |  |  |
| --- | --- | --- |
| Osteosarcoma | Recurrent infections-myelofibrosis-nephromegaly syndrome | neurodegeneration |
| spina bifida | Leukocyte adhesion deficiency type III | Usher syndrome |
| Adams-Oliver syndrome | prostate neoplasm | Male infertility with azoospermia or oligozoospermia due to single gene mutation |
| **Neoplasm** | prostate adenocarcinoma | Usher syndrome type 1 |
| Down syndrome | Hepatoerythropoietic porphyria | Male infertility with teratozoospermia due to single gene mutation |
| **cancer** | infectious disease | **body height** |
| Ataxia-telangiectasia | genetic disorder | carotid plaque build |
| lung cancer | nervous system disease | **fat body mass** |
| colorectal carcinoma | retinopathy | fatty acid measurement |
| stomach neoplasm | Retinal dystrophy | renal cell adenocarcinoma |
| hypoxia | Retinitis pigmentosa | Leber congenital amaurosis |
| colorectal neoplasm | brain disease | dementia |
| colorectal cancer | **metabolic disease** | Lewy body dementia |
| colorectal adenoma | neurodegenerative disease | synucleinopathy |
| intelligence | Genetic neurodegenerative disease | multiple system atrophy |
| red blood cell distribution width | movement disorder | blindness (disorder) |
| breast neoplasm | Neurodegeneration with brain iron accumulation | juvenile idiopathic arthritis |
| breast cancer | Pantothenate kinase-associated neurodegeneration | Mitochondrial myopathy and sideroblastic anemia |
| Leukocyte adhesion deficiency | Classic pantothenate kinase-associated neurodegeneration | Parkinson's disease |
| lymphoma | Dystonia | juvenile dermatomyositis |
| T-cell non-Hodgkin lymphoma | Cone rod dystrophy | B-cell non-Hodgkins lymphoma |

**Pathway visualization**

Se muestra la vía metabólica del cáncer de tiroides, extraída de la base de datos de [KEGG](https://www.genome.jp/kegg-bin/show_pathway?map05216) (Kyoto Encyclopedia of Genes and Genomes).

**Figura 11. *Pathway visualization* para el cáncer de tiroides. La figura muestra qué tipo de infra-expresión o sobre-expresión de genes provoca cada tipo de cáncer de tiroides.**



**Ontología de genes**

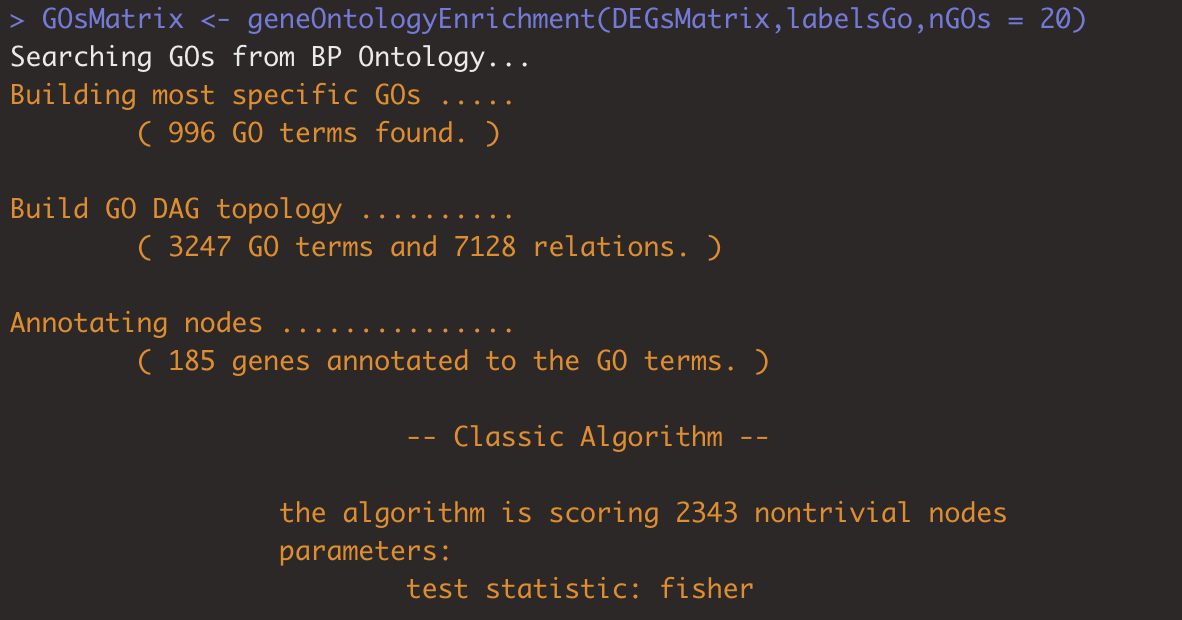
La ontología de genes ha sido descargada usando la función *geneOntologyEnrichment* de *KnowSeq.*

> labelsGo <- gsub("Solid Tissue Normal",0,labels)

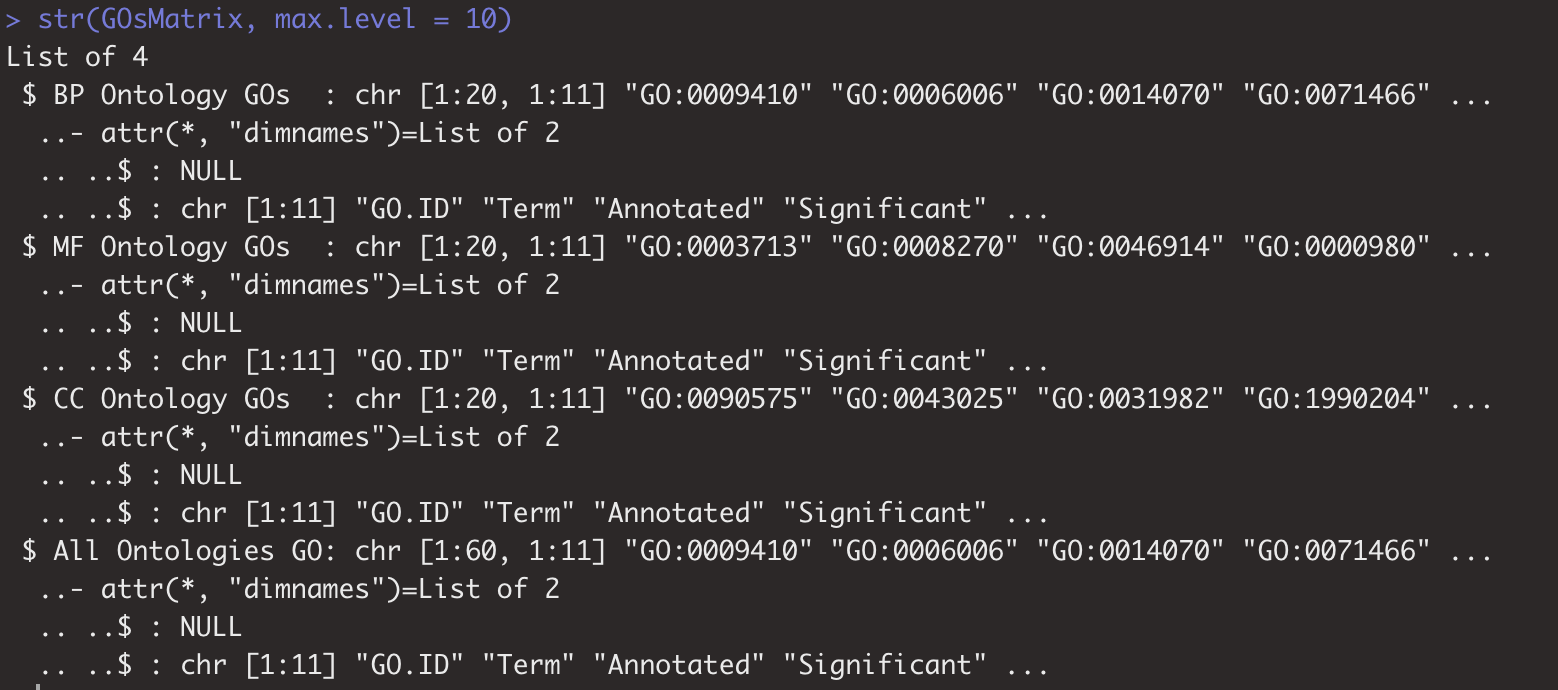
> labelsGo <- gsub("Primary Tumor",1,labelsGo)

> GOsMatrix <- geneOntologyEnrichment(DEGsMatrix,labelsGo,nGOs = 20)

**Figura 12. Captura de pantalla mostrando que se la ontología de genes se ha descargado correctamente.**



**Figura 13. Captura de pantalla mostrando que se la ontología de genes se ha descargado correctamente.**

****

**ANEXO 1: Actualización de *svm\_CV* y *knn\_CV***

Se actualiza la función *svm\_CV* para que devuelva los valores óptimos de coste y gamma del algoritmo SVM encontrados tras el tuning del modelo. La función *knn\_CV* ha sido modificada de igual forma para que devuelva el valor de *k* (número de vecinos) óptimo. A continuación se muestra la nueva función *svm\_CV*.

#' svm\_CV allows assessing the final DEGs through a machine learning step by using svm in a cross validation process.

#'

#' svm\_CV allows assessing the final DEGs through a machine learning step by using svm in a cross validation process. This function applies a cross validation of n folds with representation of all classes in each fold. The 80\% of the data are used for training and the 20\% for test. An optimization of C and G hiperparameters is done at the start of the process.

#'

#' @param data The data parameter is an expression matrix or data.frame that contains the genes in the columns and the samples in the rows.

#' @param labels A vector or factor that contains the labels for each of the samples in the data object.

#' @param vars\_selected The genes selected to classify by using them. It can be the final DEGs extracted with the function \code{\link{limmaDEGsExtraction}} or a custom vector of genes. Furthermore, the ranking achieved by \code{\link{featureSelection}} function can be used as input of this parameter.

#' @param numFold The number of folds to carry out in the cross validation process.

#' @return A list that contains five objects. The confusion matrix for each fold, the accuracy, the sensitibity and the specificity for each fold and each genes, and a vector with the best parameters found for the SVM algorithm after tuning.

#' @examples

#' dir <- system.file("extdata", package = "KnowSeq")

#' load(paste(dir, "/expressionExample.RData", sep = ""))

#'

#' svm\_CV(t(DEGsMatrix), labels, rownames(DEGsMatrix), 2)

svm\_CV <- function(data, labels, vars\_selected, numFold = 10) {

if (!is.data.frame(data) && !is.matrix(data)) {

stop("The data argument must be a dataframe or a matrix.")

}

if (dim(data)[1] != length(labels)) {

stop("The length of the rows of the argument data must be the same than the length of the lables. Please, ensures that the rows are the samples and the columns are the variables.")

}

if (!is.character(labels) && !is.factor(labels)) {

stop("The class of the labels parameter must be character vector or factor.")

}

if (is.character(labels)) {

labels <- as.factor(labels)

}

if (numFold %% 1 != 0 || numFold == 0) {

stop("The numFold argument must be integer and greater than 0.")

}

data <- as.data.frame(apply(data, 2, as.double))

data <- data[, vars\_selected]

data <- vapply(data, function(x) {

max <- max(x)

min <- min(x)

x <- ((x - min) / (max - min)) \* 2 - 1

}, double(nrow(data)))

data <- as.data.frame(data)

fitControl <- caret::trainControl(method = "cv", number = 10)

cat("Tuning the optimal C and G...\n")

grid\_radial <- expand.grid(

sigma = c(

0, 0.01, 0.02, 0.025, 0.03, 0.04,

0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.25, 0.5, 0.75, 0.9

),

C = c(

0.01, 0.05, 0.1, 0.25, 0.5, 0.75,

1, 1.5, 2, 5

)

)

dataForTunning <- cbind(data, labels)

Rsvm\_sb <- train(labels ~ ., data = dataForTunning, type = "C-svc", method = "svmRadial", preProc = c("center", "scale"), trControl = fitControl, tuneGrid = grid\_radial)

bestParameters <- c(cost = Rsvm\_sb$bestTune$C, gamma = Rsvm\_sb$bestTune$sigma)

cat(paste("Optimal cost:", bestParameters[1], "\n"))

cat(paste("Optimal gamma:", bestParameters[2], "\n"))

acc\_cv <- matrix(0L, nrow = numFold, ncol = dim(data)[2])

sens\_cv <- matrix(0L, nrow = numFold, ncol = dim(data)[2])

spec\_cv <- matrix(0L, nrow = numFold, ncol = dim(data)[2])

cfMatList <- list()

for (i in seq\_len(numFold)) {

cat(paste("Training fold ", i, "...\n", sep = ""))

trainingDataset <- setNames(

data.frame(matrix(ncol = ncol(data), nrow = 0)),

colnames(data)

)

testDataset <- setNames(

data.frame(matrix(ncol = ncol(data), nrow = 0)),

colnames(data)

)

labelsTrain <- factor(0L)

labelsTest <- factor(0L)

for (class in names(table(labels))) {

classPos <- which(labels == class)

classPos <- sample(classPos)

trainingPos <- round(length(classPos) \* 0.8)

testPos <- round(length(classPos) \* 0.2)

trainingDataset <- rbind(trainingDataset, data[classPos[seq\_len(trainingPos)], ])

testDataset <- rbind(testDataset, data[classPos[(trainingPos + 1):(trainingPos + testPos)], ])

labelsTrain <- unlist(list(labelsTrain, labels[classPos[seq\_len(trainingPos)]]))

labelsTest <- unlist(list(labelsTest, labels[classPos[(trainingPos + 1):(trainingPos + testPos)]]))

}

labelsTrain <- factor(labelsTrain[-1])

labelsTest <- factor(labelsTest[-1])

for (j in seq\_len(length(vars\_selected))) {

svm\_model <- svm(trainingDataset[, seq(j)], labelsTrain,

kernel = "radial",

cost = Rsvm\_sb$bestTune$C, gamma = Rsvm\_sb$bestTune$sigma, probability = TRUE

)

predicts <- predict(svm\_model, testDataset[, seq(j)], probability = TRUE)

cfMatList[[i]] <- confusionMatrix(predicts, labelsTest)

acc\_cv[i, j] <- confusionMatrix(predicts, labelsTest)$overall[[1]]

sens\_cv[i, j] <- confusionMatrix(predicts, labelsTest)$byClass[[1]]

spec\_cv[i, j] <- confusionMatrix(predicts, labelsTest)$byClass[[2]]

}

}

rownames(acc\_cv) <- paste("Fold", seq(numFold), sep = "")

colnames(acc\_cv) <- vars\_selected

rownames(sens\_cv) <- paste("Fold", seq(numFold), sep = "")

colnames(sens\_cv) <- vars\_selected

rownames(spec\_cv) <- paste("Fold", seq(numFold), sep = "")

colnames(spec\_cv) <- vars\_selected

cat("Classification done successfully!\n")

results\_cv <- list(cfMatList, acc\_cv, sens\_cv, spec\_cv, bestParameters)

names(results\_cv) <- c("cfMats", "accMatrix", "sensMatrix", "specMatrix", "bestParameters")

invisible(results\_cv)

}

**ANEXO 2: Actualización de *dataPlot***

Se ha actualizado la función *dataPlot* para que en los gráficos de validación cruzada, cada fold se pinte con una línea discontinua en lugar de sólida, para intentar distinguir mejor las líneas. No se presenta la función completa, sólo la parte que se ha modificado (**en negrita**).

...

if(is.matrix(data)){

plot(data[1,],type='l',col=colours[1], main=main, **lty = "dashed"**,

xlab=xlab,ylab=ylab,axes=TRUE,frame.plot=TRUE,lwd = 2.5, ylim = c(min(data),max(data)))

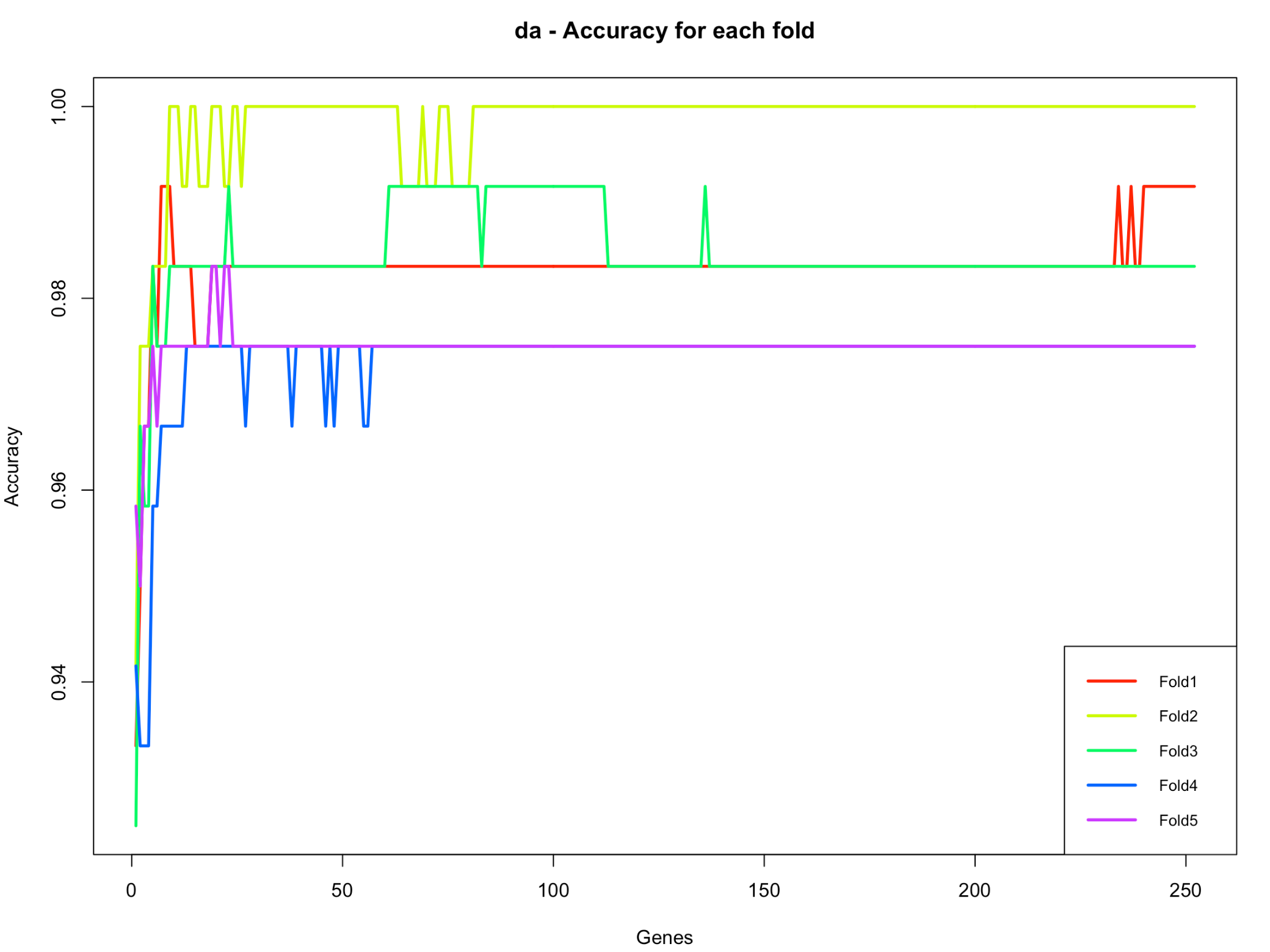
for(i in c(2:dim(data)[1])){

lines(data[i,],col=colours[i], **lty="dashed"**,lwd = 2.5)

}

...

(Antes)



(Ahora)

