

Análisis de expresión diferencial de genes y reconocimiento de grupos mutados de cáncer colorrectal en México con Machine Learning

Instituto Tecnológico y de Estudios Superiores de Monterrey

Escuela de ingeniería y ciencias

**Abraham Cepeda
Oseguera**

a00827666@itesm.mx

**Juan Carlos Garfias
Tovar**

a01652138@itesm.mx

**Lucas Eduardo Idígoras
Laitano**

a00827751@itesm.mx

Vanina Sada Gonzalez

a00827037@itesm.mx

1. Resumen

Este reporte científico analiza datos respecto a personas con cáncer de colon en México, así como bases de datos que proporcionan la expresión diferencial de un gran número de genes en el tejido de colon sano e infectado de cáncer. De modo que, se busca encontrar relaciones entre los genes para poder determinar un método capaz de diagnosticar si una persona puede llegar a contraer este tipo de cáncer. Igualmente, se buscan posibles tratamientos de acuerdo a genes específicos. Agrupando las mutaciones de cada sector con algoritmos de machine learning.

2. Introducción

El cáncer de colon ha tenido un aumento en casos entre adultos jóvenes en los últimos años. Solía ser que las personas mayores de 50 años eran más propensas a tenerlo. Por lo general, el cáncer de colon aparece como pequeños grupos no cancerosos (benignos) de células llamadas pólipos que se forman en el interior del colon. Con el tiempo, algunos de estos pólipos pueden convertirse

en cánceres de colon. Hay una relación entre personas con cáncer de colon, y cáncer rectal ya que el colon y el rectal están en cercana proximidad. Es por eso que los términos de cáncer de colon, cáncer rectal, y cáncer colorrectal se usan indistintamente.

Para este proyecto, analizamos una variedad de información y también la programamos en R para obtener resultados, y luego los analizamos con el objetivo final de comprender varios factores. Algunos de ellos, lo que está provocando este cáncer entre las personas más jóvenes, existen ciertos factores de riesgo que podríamos prevenir, o si es genético. También queríamos saber qué causa el cáncer y otras enfermedades, y cómo la ciencia hoy nos ha ayudado a atacar estos problemas. Todos para saber cómo la ciencia de los datos y los nuevos descubrimientos en bioinformación pueden ayudarnos.

3. Genes relacionados al cáncer de colon

El gen MLH1 es un conjunto de genes de reparación de desajuste. Este gen le da

instrucciones a la proteína y toma un rol importante cuando se repara un pedazo de ADN. Esta proteína ayuda a corregir errores producidos cuando se copia el ADN en preparación para la división celular. La proteína MLH1 se une con otra proteína PMS2 y forman un complejo llamado dímero. Juntas coordinan las actividades de otras proteínas que reparan los errores cometidos cuando se replica el ADN. Se elimina una sección del ADN que tiene errores, y se reemplaza la sección con una secuencia corregida.

El gen MSH2 tiene un rol parecido al MLH1. Este gen proporciona instrucciones para hacer una proteína, y igual que MLH1 tiene un rol esencial en la reparación del ADN. Esta proteína ayuda a corregir los errores que se producen cuando se copia el ADN en preparación para la división celular. Esta proteína se une con MSH6 o MSH3 para formar un complejo de dos proteínas llamado dímero. Este complejo identifica las ubicaciones en el ADN donde hay errores que se cometieron cuando se replicó.

Han habido estudios en los que se ha visto que más pacientes sobreviven con cáncer colorrectal hereditario asociado a MLH1. Se hizo otro estudio para identificar variaciones fenotípicas en la expresión de cáncer colorrectal entre familias de cáncer colorrectal hereditario sin poliposis (Síndrome de Lynch) con mutaciones de la línea germinal MLH1 y MSH2. Las personas con síndrome de Lynch es un tipo de síndrome de cáncer hereditario asociado con una predisposición genética a diferentes tipos de cáncer. Se compararon las tasas de

cáncer sincrónico, metacrónico, estadio tumoral incidencia de cáncer extra colónico y supervivencia, entre otras cosas. Se determinó que la presencia de cáncer rectal no debería de impedir el diagnóstico de HNPCC (Síndrome de Lynch), porque la incidencia de cáncer rectal en MSH2 fue comparable con la de la población general. Las variaciones fenotípicas no resultaron en diferencias entre subgrupos genotípicos. Estas características fenotípicas de los genotipos de HNPCC pueden tener importancia clínica en el diseño de pruebas de detección, vigilancia, y seguimiento específicos para personas afectadas.

El síndrome de Lynch es una enfermedad genética en que existe un mayor riesgo de tener cáncer. El síndrome de Lynch se hereda de forma autosómica dominante. Esto significa que un rasgo o trastorno se puede transmitir de padres a hijos. Si el hijo hereda el gen anormal de solo uno de los padres, pueden presentar la enfermedad.

Las personas que tienen el síndrome de Lynch tienen un mayor riesgo de tener cáncer de varios tipos como de colon, recto, estómago, intestino delgado, hígado, vesícula, vías urinarias, cerebro, piel, y próstata. Hay tendencias entre mujeres con síndrome de Lynch que pueden tener cáncer de ovarios o de útero.

4. Análisis de pacientes de nuevo ingreso en México

Igualmente, se utilizaron los datos del Instituto Nacional de Cancerología (INCAN), los cuales contienen información

relevante respecto a los nuevos casos de cáncer en nuestro país. Por lo que, estos datos se pueden filtrar de tal forma que se pueden analizar los casos de cáncer de colon en México.

Para comenzar, se almacenaron los datos en un *dataframe* para luego a filtrarlos con el objetivo de tener otro *dataframe* que contenga solamente la información recopilada de los pacientes que sufren de cáncer de colon como se muestra en la Figura 1.

```
1 # <-----Load data----->
2 data <- read.csv("nuevo_ingreso.csv")
3 data_colon <- dataaa[dataaa$DESCRIPCION_DIAGNOSTICO == "TUMOR MALIGNO DEL COLON",]
```

Figura 1. Código de almacenamiento y filtrado de datos.

Una vez que ya se tenía un *dataframe* con toda la información necesaria, se realizaron diversos filtros para encontrar informaciones y relaciones relevantes.

En primera instancia, se calculó el rango de edades de las personas que cuentan con este tipo de cáncer mediante el código mostrado en la Figura 2. El resultado obtenido fue de 19 a 86 años como se muestra en la Figura 3.

```
6 # <-----Rango de edad----->
7 cat("Rango de edad: ",range(data_colon$EDAD))
```

Figura 2. Código para calcular rango de edad e imprimirlo en la consola.

```
> cat("Rango de edad: ",range(data_colon$EDAD))
Rango de edad: 19 86>
```

Figura 3. Resultado obtenido tras correr el código de la Figura 2.

Después, se compararon los casos entre hombres y mujeres en general con el código de la Figura 4 y los resultados obtenidos se

muestran en la Figura 5. Como se muestra en la figura, mayormente los casos son hombres. Sin embargo, la diferencia entre casos de hombres y mujeres no es tan significativa.

```
9 # <-----Número de mujeres----->
10 data_mujeres <- data_colon[data_colon$SEXO == "Femenino",]
11 casos_mujeres <- length(data_mujeres[[1]])
12
13 # <-----Número de hombres----->
14 data_hombres <- data_colon[data_colon$SEXO == "Masculino",]
15 casos_hombres <- length(data_hombres[[1]])
```

Figura 4. Código para obtener el número de casos de hombres y mujeres.

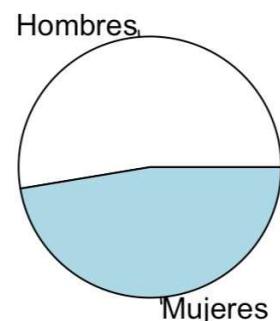


Figura 5. Resultado obtenido tras analizar el número de casos de hombres y mujeres de manera general.

Posteriormente, se obtuvieron los casos en personas jóvenes y personas mayores. Se tomaron en cuenta a las personas menores a 50 años como menores, ya que, comúnmente, los artículos que analizan el cáncer de colon, indican que este tipo de cáncer normalmente se da cuando una persona es mayor a 50 años. Además de comparar estos casos, se analizó la relación entre hombres y mujeres dentro del grupo de los jóvenes, así como de los mayores.

Los resultados obtenidos tras correr el código de la Figura 6 se muestran en las Figuras 7, 8 y 9. La Figura 7 demuestra que el cáncer de colon es considerablemente más común en personas mayores que jóvenes.

Sin embargo, debido a que la diferencia no es tan grande, resulta empírico conocer la razón por la cual se enferman las personas jóvenes. Por otro lado, la Figura 8 indica que dentro de las personas mayores, las mujeres se enferman un poco más pero en un porcentaje bajo. Por lo que, se puede decir que el sexo no resulta un factor que determine la probabilidad de padecer de cáncer de colon a una edad avanzada. Sin embargo, la Figura 9, claramente demuestra los hombres jóvenes se enferman de este tipo de cáncer más frecuentemente que las mujeres jóvenes.

```

18 # <-----Número de jóvenes----->
19 data_jovenes <- data_colon[data_colon$EDAD < 50,]
20 casos_jovenes <- length(data_jovenes[[1]])]
21
22 # <-----Número de mujeres jóvenes----->
23 data_mujeres_jovenes <- data_jovenes[data_jovenes$SEXO == "Femenino",]
24 casos_mujeres_jovenes <- length(data_mujeres_jovenes[[1]])]
25
26 # <-----Número de hombres jóvenes----->
27 data_hombres_jovenes <- data_jovenes[data_jovenes$SEXO == "Masculino",]
28 casos_hombres_jovenes <- length(data_hombres_jovenes[[1]])]
29
30 # <-----Número de mayores----->
31 data_mayores <- data_colon[data_colon$EDAD >= 50,]
32 casos_mayores <- length(data_mayores[[1]])]
33
34 # <-----Número de mujeres mayores----->
35 data_mujeres_mayores <- data_mayores[data_mayores$SEXO == "Femenino",]
36 casos_mujeres_mayores <- length(data_mujeres_mayores[[1]])]
37
38 # <-----Número de hombres mayores----->
39 data_hombres_mayores <- data_mayores[data_mayores$SEXO == "Masculino",]
40 casos_hombres_mayores <- length(data_hombres_mayores[[1]])]
```

Figura 6. Código para obtener el número de casos de personas jóvenes y mayores, así como de hombres y mujeres en cada grupo

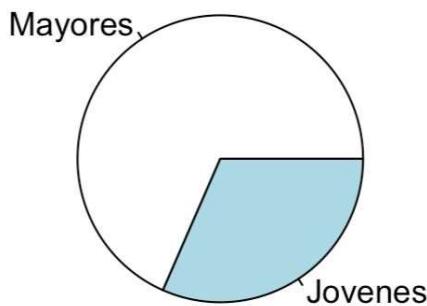


Figura 7. Resultado obtenido tras analizar el número de casos personas jóvenes y mayores.



Figura 8. Resultado obtenido tras analizar el número de casos de hombres y mujeres mayores.

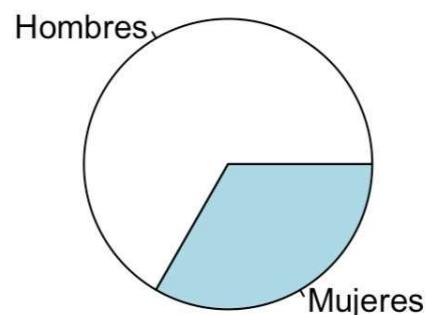


Figura 9. Resultado obtenido tras analizar el número de casos de hombres mayores y mujeres mayores.

5. Análisis de expresión diferencial de genes en el tejido de colon

A parte de analizar los datos respecto a los pacientes de cáncer de nuevo ingreso en México, se empleó la base de datos utilizada para un artículo publicado en el *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, el cual se titula “Multiclass cancer diagnosis using tumor gene expression signatures.” Los datos recopilados por los autores de este artículo, contienen la expresión de 15,970 genes en diferentes tipos de tejidos tanto de personas con tumores cancerígenos como de personas sanas. Debido a que uno de los tejidos de consideraron es el colon, se

pudieron utilizar estos datos para encontrar relaciones o discrepancias entre los genes expresados en el colon de una persona con cáncer de colon y de una persona sana.

Similar al análisis anterior, el primer paso para comenzar a manipular y analizar los datos es importarlos al programa, lo cual se logró con el código que muestra la Figura 10.

```
1 # <-----Load data----->
2 load("Multi_Cancer_Data.RData", verbose = TRUE)
3 df <- data.frame(multi_cancer_data)
```

Figura 10. Código de importación y almacenamiento de datos.

Tras importar los datos a *RStudio* se notó que los títulos de las columnas tenían cierto patrón pero que era claro que las columnas que pertenecían a cáncer colorrectal tenían nombres similares pero no exactos. Por ejemplo, las columnas de los tumores comenzaban con “Tumor” luego venía el tipo de tumor y, finalmente, algún tipo de especificación de la columna. Por lo que, para poder analizar únicamente las columnas que correspondían a cáncer de colon, se tuvo que renombrar todas las columnas de tal modo que solamente incluyeran el tipo de cáncer. De modo que se empleó el código de la Figura 11 para renombrar todas las columnas.

```
5 # <-----Cambiar nombre de columnas----->
6 # <----Casos tumor---->
7 for(i in 1:190) {
8   name <- colnames(df)[i]
9   name <- gsub("Tumor_","",name)
10  names<-strsplit(name, '_')
11  name <- names[[1]][1]
12  colnames(df)[i] <- name
13 }
14 # <----Casos normal---->
15 for (i in 191:280) {
16   name <- colnames(df)[i]
17   name <- gsub("Normal_","",name)
18   names<-strsplit(name, '_')
19   name <- names[[1]][1]
20   colnames(df)[i] <- name
21 }
```

Figura 11. Código para renombrar columnas de base de datos.

Una vez que ya se tenían las columnas nombradas correctamente, se identificó que las columnas que contenían la información de los casos de tumor en el colon contenían el título de “Colorrectal,” mientras que los datos del tejido de colon en personas sanas tenía el título de “Colon.” Al haber identificado este hecho, se escribió el código de la Figura 12 para tener dos *dataframes*. Uno que contuviera las columnas de los casos con cáncer de colon y otro con la expresión de genes en el colon en personas sanas.

```
23 # <-----Crear dataframes----->
24 # <----Dataframes iniciales---->
25 df_tumor <- df[,which(colnames(df)=="Colorrectal")]
26 df_normal <- df[,which(colnames(df)=="Colon")]
```

Figura 12. Código para crear los dos *dataframes* necesarios.

Posteriormente, se crearon dos *dataframes* adicionales, con el objetivo de almacenar los promedios de cada gen tanto en el caso de tejido infectado como sano. Para optimizar el algoritmo, inicialmente, se declararon los *dataframes* con valores “NA” y con las dimensiones adecuadas como se muestra en la Figura 13. Más adelante se obtuvieron los promedios por cada gen y se almacenaron en el *dataframe* correspondiente, lo cual se observa claramente en la Figura 14. Una vez calculados los promedios de cada gen, se

deben de ordenar los datos de manera decreciente con el propósito de poder almacenar los 20 genes más expresados en dos nuevos *dataframes*, como se muestra en la Figura 15.

```
28 # <----Dataframes de promedios----->
29 v_tumor <- data.frame(row.names(df_tumor), rep(NA, length(df_tumor[[1]])))
30 v_normal <- data.frame(row.names(df_normal), rep(NA, length(df_tumor[[1]])))
```

Figura 13. Código para inicializar dos nuevos *dataframes* que almacenan los promedios de cada gen.

```
32 # <-----Calcular y almacenar promedios----->
33 for (i in 1:length(df_tumor[[1]])) {
34   v_tumor[[2]][i] <- mean(as.numeric(df_tumor[i,]))
35   v_normal[[2]][i] <- mean(as.numeric(df_normal[i,]))
36 }
```

Figura 14. Código para calcular los promedios por gen y almacenarlos en sus respectivos *dataframes*.

```
38 # <-----Ordenar de mayor a menor los promedios----->
39 v_tumor <- v_tumor[order(v_tumor$Promedio, decreasing = TRUE),]
40 v_normal <- v_normal[order(v_normal$Promedio, decreasing = TRUE),]
41
42 # <-----Obtener los 20 mayores promedios----->
43 twenty_tumor <- v_tumor[1:20,]
44 twenty_normal <- v_normal[1:20,]
```

Figura 15. Código para ordenar los *dataframes* que contienen los promedios y almacenar los 20 mayores en dos nuevos *dataframes*.

Al correr los códigos de la Figura 10 a la 15, se obtuvieron dos *dataframes*. Uno con los 20 promedios más altos de la expresión de genes en el tejido de colon infectado de cáncer. El segundo, igualmente, con los 20 promedios más altos de la expresión de genes pero en tejido de colon sano. Los resultados de estos *dataframes* se pueden observar en las Figuras 16 y 17 respectivamente.

	Gene	Promedio
1844	CARCINOEMBRYONIC ANTIGEN PRECURSOR_M29540_at	2.538545
1352	MMP12 Matrix metalloproteinase 12 (macrophageela..._at	2.307545
3279	CDX1 Caudal type homeo box transcription factor 1_..._at	2.144182
4209	MMP1 Matrix metalloproteinase 1 (interstitial collagen..._at	2.088818
1924	TUMOR-ASSOCIATED ANTIGEN CO-029_M35252_at	2.008545
165	GC-Box binding protein BTEB2_D14520_at	1.775000
2176	Transforming growth factor-beta induced gene produ..._at	1.732727
3341	Gamma-glutamyl hydrolase (hGH) mRNA_U55206_at	1.710818
12197	NF-E2-related factor 3_RC_AA132523_at	1.659273
11927	EST: z174e07.s1 Stratagene colon (#937204) Homo sa..._at	1.617545
5431	LI-cadherin_X83228_at	1.594727
4066	VILLIN_X12901_at	1.584091
3280	Homeobox protein Cdx2 mRNA_U51096_at	1.581818
4013	MMP3 Stromelysin_X05232_at	1.561000
1536	PLCB4 Phospholipase C, beta 4_L41349_at	1.545364
6304	KRT8 Keratin 8_X74929_s_at	1.506273
4409	GPX2 Glutathione peroxidase 2, gastrointestinal_X68..._at	1.501545
12933	EST: zr77g09.s1 Soares NHMPu S1 Homo sapiens cD..._at	1.499636
1711	NCA Non-specific cross reacting antigen_M18728_at	1.461091
12800	Phospholipase C, beta 4_RC_AA242819_at	1.446000

Figura 16. *Dataframe* con los 20 genes con mayor expresión en el tejido de colon infectado de cáncer.

	Gene	Promedio
1607	HBG2 Hemoglobin gamma-G_M10050_at	3.812000
1131	DRA Down-regulated in adenoma_L02785_at	3.630545
5460	Galectin-4_AB006781_s_at	3.618000
13344	Selenium binding protein 1_RC_AA290679_at	3.533909
5431	LI-cadherin_X83228_at	3.400364
4832	Carcinoembryonic antigen family member 2, CGM2_X..._at	3.046818
6408	Intestinal peptide-associated transporter HPT-1 mRN..._at	2.996091
3279	CDX1 Caudal type homeo box transcription factor 1_..._at	2.916909
14726	Human N-benzoyl-L-tyrosyl-p-amino-benzoic acid h..._at	2.913636
11568	EST: zh89b04.s1 Soares fetal liver spleen 1NFLS S1 H..._at	2.911545
573	IgG Fc binding protein_D84239_at	2.910182
9301	IgG Fc binding protein_D84239_at-2	2.910182
13695	EST: zv63a12.s1 Soares total fetus Nb2HF8 9w Homo ..._at	2.866000
4644	Clone HSH1 HMG CoA synthase mRNA, partial cds_X8..._at	2.856455
4122	BGP Biliary glycoprotein (alternative products)_X1635..._at	2.847091
5751	Vasoactive Intestinal Peptide_HG2987-HT3136_s_at	2.818182
12980	Homo sapiens mRNA expressed in thyroid gland_RC_..._at	2.811273
5251	MUC2 Mucin 2, intestinal/tracheal_L21998_at	2.658273
1892	SRI Sorcin_M32886_at	2.640909
3740	A33 antigen precursor mRNA_U79725_at	2.617273

Figura 17. *Dataframe* con los 20 genes con mayor expresión en el tejido de colon sano.

Ya que se tenían los 20 genes más expresados en cada caso, resulta lógica la cuestión de cuáles genes son comunes, lo

cual se logró con el código de la Figura 18. Este código inicializa un *dataframe* de tres columnas, ya que se requiere conocer el nombre del gen, su expresión en el caso de tumor y su expresión en el caso de normal. Además, este *dataframe* cuenta con una longitud inicial de 20 filas, puesto que el máximo número de similitudes es este número. Una vez declarado el *dataframe*, se empleó el código de la Figura 19 para determinar cuáles de los 20 genes de tumor se encuentran dentro de los 20 genes de los normales. Tras hacer este análisis se implementó la función *na.omit*, mostrada en la Figura 20, para remover las filas del *dataframe* que no adquirieron ningún valor.

```
46 # <-----Dataframe de genes iguales----->
47 equal <- data.frame(rep(NA,20),rep(NA,20),rep(NA,20))
48 colnames(equal) <- c("Gene", "Tumor", "Normal")
```

Figura 18. Código inicializar *dataframe* que contiene los genes que aparecen en ambos casos (tumor y normal).

```
50 # <-----Determinar los genes iguales----->
51 count <- 1
52 for (i in 1:20) {
53   if (twenty_tumor[i,1] %in% twenty_normal[,1]){
54     equal[count,1] <- as.character(twenty_tumor[i,1])
55     equal[count,2] <- twenty_tumor[,2]
56     equal[count,3] <- twenty_normal[twenty_normal$Gene == as.character(twenty_tumor[i,1]),2]
57     count <- count + 1
58   }
59 }
```

Figura 19. Código determinar los genes que aparezcan en ambos *dataframes* (tumor y normal).

```
61 # <-----Remover NAs----->
62 equal <- na.omit(equal)
```

Figura 20. Código para remover los valores vacíos del *dataframe* “equal.”

Los resultados obtenidos tras determinar cuáles genes coinciden en ambos casos se muestran en la Figura 21. Como se puede observar, solamente dos genes se encuentran

dentro de los 20 promedios mayores de ambos tipos de dato.

	Gene	Tumor	Normal
1	CDX1 Caudal type homeo box transcription factor 1_...	2.144182	2.916909
2	LI-cadherin_X83228_at	1.594727	3.400364

Figura 21. *Dataframe* con los dos genes que se encuentran dentro de los *dataframes* anteriores.

Otras forma de analizar los datos recopilados hasta el momento, es revisando la expresión que tienen los 20 genes con mayor promedios de los tumores en los datos normales y viceversa. De modo que, similarmente a pasos anteriores, se inicializan los *dataframes* necesarios para almacenar la información pertinente, lo cual se hizo con el código de la Figura 22. Una vez declarados los *dataframes* se desarrolló el código de la Figura 23 para hacer las comparaciones. Igualmente, se incluyó el código de la Figura 24 para ordenar los datos recopilados de acuerdo a la diferencias como se muestra en los resultados obtenidos tras correr el código que se encuentran en las Figuras 25 y 26.

```
64 # <-----Dataframe de comparacion de genes----->
65 tumor_in_normal <- data.frame(twenty_tumor[,1],rep(NA,20),rep(NA,20),rep(NA,20))
66 colnames(tumor_in_normal) <- c("Gene", "Normal", "Tumor", "Diferencia")
67 normal_in_tumor <- data.frame(twenty_normal[,1],rep(NA,20),rep(NA,20),rep(NA,20))
68 colnames(normal_in_tumor) <- c("Gene", "Tumor", "Normal", "Diferencia")
```

Figura 22. Código para ordenar el *dataframe* de mayor a menor.

```
70 # <-----Realizar comparacion y almacenar datos----->
71 for (i in 1:20) {
72   tumor_in_normal[i,2] <- v_normal[v_normal$Gene == twenty_tumor[i,1],2]
73   tumor_in_normal[i,3] <- twenty_tumor[i,2]
74   tumor_in_normal[i,4] <- abs(twenty_tumor[i,2]-tumor_in_normal[i,2])
75   normal_in_tumor[i,2] <- v_tumor[v_tumor$Gene == twenty_normal[i,1],2]
76   normal_in_tumor[i,3] <- twenty_normal[i,2]
77   normal_in_tumor[i,4] <- abs(twenty_normal[i,2]-normal_in_tumor[i,2])
78 }
```

Figura 25. Código para almacenar los datos correspondientes de cada gen, así como las diferencias entre los promedios de tumor y normal.

```

80 # <-----Ordenar los genes de mayor a menor----->
81 tumor_in_normal <- tumor_in_normal[order(tumor_in_normal$Diferencia, decreasing = TRUE),]
82 normal_in_tumor <- normal_in_tumor[order(normal_in_tumor$Diferencia, decreasing = TRUE),]

```

Figura 24. Código para ordenar el *dataframe* de mayor a menor.

	Gene	Normal	Tumor	Diferencia
4	MMP1 Matrix metalloproteinase 1 (interstitial collagen... 1	-0.218454545	0.208818	2.30727273
2	MMP12 Matrix metalloproteinase 12 (macrophage ela... 7	0.005090909	2.307545	2.30245455
9	Transforming growth factor-beta induced gene produ... 11	-0.157181818	1.732727	1.88990909
11	LI-cadherin_X83228_at	3.400363636	1.594727	1.80563636
14	MMP3 Stromelysin_X05232_at	0.123090909	1.561000	1.43790909
15	PLCB4 Phospholipase C, beta 4_L41349_at	0.175272727	1.545364	1.37009091
8	Gamma-glutamyl hydrolase (hGH) mRNA_U55206_at	0.503000000	1.710818	1.20781818
19	NCA Non-specific cross reacting antigen_M18728_at	0.272363636	1.461091	1.18872727
20	Phospholipase C, beta 4_RC_AA242819_at	0.543090909	1.446000	0.90290909
18	EST: zr77g09.s1 Soares NhHMPu S1 Homo sapiens cD... 3	2.367545455	1.499636	0.86790909
3	CDX1 Caudal type homeo box transcription factor 1_... 13	2.916909091	2.144182	0.77272727
13	Homeobox protein Cdx2 mRNA_U51096_at	0.887181818	1.581818	0.69463636
12	VILLIN_X12901_at	0.942363636	1.584091	0.64172727
1	CARCINOEMBRYONIC ANTIGEN PRECURSOR_M29540_at	2.060636364	2.538545	0.47790909
10	EST: zl74e07.3.1 Stratagene colon (#937204) Homo sa... 5	1.913090909	1.617545	0.29554545
5	TUMOR-ASSOCIATED ANTIGEN CO-029-M35252_at	2.278181818	2.008545	0.26963636
17	GPX2 Glutathione peroxidase 2, gastrointestinal_X68... 6	1.269454545	1.501545	0.23209091
6	GC-Box binding protein BTB2_D14520_at	1.946363636	1.775000	0.17136364
16	KRT8 Keratin 8_X74929_s_at	1.464909091	1.506273	0.04136364

Figura 25. *Dataframe* con la comparación de los 20 genes más expresados en el caso de tumor y su expresión en el tejido normal.

	Gene	Tumor	Normal	Diferencia
4	Selenium binding protein 1_RC_AA290679_at	0.129818182	3.533909	3.4040909
2	DRA Down-regulated in adenoma_I02785_at	0.392090909	3.630545	3.2384545
1	HBG2 Hemoglobin gamma-G_M10050_at	0.723636364	3.812000	3.0883636
10	EST: zh89b04.s1 Soares fetal liver spleen INFSL S1 H... 16	-0.175090909	2.911545	3.0866364
16	Vasoactive Intestinal Peptide_HG2987-HT3136_s_at	-0.175636364	2.818182	2.9938182
11	IgG Fc binding protein_D84239_at	0.005363636	2.910182	2.9048182
12	IgG Fc binding protein_D84239_at-2	0.005363636	2.910182	2.9048182
6	Carcinoembryonic antigen family member 2, CGM2_X... 14	0.234000000	3.046818	2.8128182
14	Clone HSH1 HMG CoA synthase mRNA, partial_cds_X8... 17	0.522727273	2.856455	2.3337273
17	Homo sapiens mRNA expressed in thyroid gland_RC... 3	0.481818182	2.811273	2.3294545
3	Galectin-4_AB006781_s_at	1.309636364	3.618000	2.3083636
13	EST: zv63a12.s1 Soares total fetus Nb2HF8 9w Homo ... 15	0.574363636	2.866000	2.2916364
15	BGP Biliary glycoprotein [alternative products]_X1635... 20	0.561636364	2.847091	2.2854545
20	A33 antigen precursor mRNA_U79725_at	0.400090909	2.617273	2.2171818
19	SRI Sorcin_M32886_at	0.559181818	2.640909	2.0817273
9	Human N-benzoyl-L-tyrosyl-p-amino-benzoic acid h... 5	0.887181818	2.913636	2.0264545
5	LI-cadherin_X83228_at	1.594727273	3.400364	1.8056364
7	Intestinal peptide-associated transporter HPT-1 mRN... 18	1.356454545	2.996091	1.6396364
18	MUC2 Mucin 2, intestinal/tracheal_L21998_at	1.315454545	2.658273	1.3428182
8	CDX1 Caudal type homeo box transcription factor 1_... 1	2.144181818	2.916909	0.7727273

Figura 26. *Dataframe* con la comparación de los 20 genes más expresados en el caso de tejido normal y su expresión en el tejido de tumor.

6. Posibles Tratamientos del CCR

Uno de los tratamientos que se ha mostrado más efectivo a la hora de tratar cánceres sólidos, incluyendo el cáncer colorrectal es la inmunoterapia. A diferencia de los otros métodos para tratar el cáncer metastásico la inmunoterapia logra una remisión de largo plazo haciéndolo una de las mejores opciones para tratar el CCR metastásico.

Otro tratamiento al cual la FDA le aceleró el proceso de aprobación es el Tratamiento con inhibidor del punto de control inmunitario (ICI). Son anticuerpos que enfrentan la muerte celular programada para que no se sigan acomunando células muertas. Esto a mejorado el índice de supervivencia del CCR.

Además de los tratamientos posibles para el CCR también se implementan medidas con los pacientes antes de una operación quirúrgica para maximizar las posibilidades de recuperación después de la operación. Los elementos principales que se implementan para aumentar el éxito de la cirugía son:

- Ejercicio preoperatorio
- Terapia nutricional
- Técnicas de reducción de ansiedad

De acuerdo con la doctora Satya Das, una metastasectomía puede ser una opción para pacientes con baja carga de enfermedad y metástasis hepáticas o pulmonares accesibles.

7. Análisis de expresión diferencial

El TCGA es una librería permite identificar el conjunto completo de cambios en el ADN. Con esto es posible entender cómo están formados a nivel genético los tipos de cáncer. En este caso con el fin de comprender la formación, detonadores y consecuencias del cáncer se aplicó un análisis de expresión diferencial a partir del cancer colorrectal para dos grupos, el primero orientado a adultos jóvenes y el segundo para adultos hasta adultos mayores. El análisis de expresión diferencial está orientado para conocer y localizar genes que se expresan diferente entre los dos grupos, permitiendo ver en qué se diferencian las muestras.

Como primer paso se importa el archivo con todos los datos genéticos. Se crean variables para almacenar todos los índices de jóvenes y adultos junto con contadores para ambos.

```

7 | library(gtools)
8 | library(dplyr)
9 |
10| load("TCGA_COADREAD_comp_data.RData", verbose = TRUE)
11| df <- data.frame(tcga_coadread)
12| type <- data.frame(tcga_coadread_class)
13| type <- as.vector(type[,1])
14| yindex <- vector()
15| oindex <- vector()
16|
17| #conteo de tipos de cancer
18| nyoung<-0
19| nold<-0

```

Figura 27. Código base para lectura de base de datos y declaración de variables para grupos

Una vez cargado el archivo se itera sobre los índices a partir del data frame type para obtener las posiciones de cada tipo de cáncer.

```

21| #recolección de indices
22| counter1 =1
23| counter2 =1
24| for(i in 1:length(type)){
25|   if(type[i]=="Young"){
26|     nyoung<-nyoung+1
27|     yindex[counter1]=i
28|     counter1=counter1+1
29|   }else{
30|     nold<-nold+1
31|     oindex[counter2]=i
32|     counter2=counter2+1
33|   }
34| }

```

Figura 28. Código base para recolección de índices y conteo de elementos de grupos

Posteriormente se crean dos dataframes para cada tipo de cáncer y a partir de ello se hace un t test recolectando los p values para cada gen a partir de iteraciones sobre el número de genes en los datos.

```

37| #T Test
38| young <- df[,yindex]
39| old <- df[,oindex]
40|
41| pvalues <-data.frame(matrix(, nrow=length(df[,1]), ncol=1))
42|
43| for(i in 1:nrow(df)){
44|   pvalues[i,1]<-t.test(df[i,yindex],df[i,oindex])$p.value
45| }
46| colnames(pvalues)<-c("Pvalues")
47| rownames(pvalues)<-rownames(df)

```

Figura 29. Código base para realización de prueba estadística t-test para cada gen a partir de los índices de cada grupo

Una vez obtenidos estos datos se obtienen los promedios para cada uno de los grupos para así posteriormente compararlos y realizar histogramas. Además de esto se hace un filtrado de los p values menores a 0.05, dejando únicamente aquellos que tienen una diferencia estadística significativa entre los dos grupos.

```

50 #Getting mean graphs
51 type <- data.frame(tcga_coadread_class)
52 Averages <- data.frame(matrix(0, nrow = length(df[[1]]), ncol = 2))
53 countYoung <- 0
54 countOld <- 0
55 for(i in 1:length(df)) {
56   if(type[[1]][i] == "Young"){
57     Averages[[1]][i] = Averages[[1]][i] + df[[i]]
58     countYoung <- countYoung + 1
59   } else{
60     Averages[[2]][i] = Averages[[2]][i] + df[[i]]
61     countOld <- countOld + 1
62   }
63 }
64 for(i in 1:length(Averages[[1]])){
65   Averages[[1]][i] = Averages[[1]][i]/countYoung
66   Averages[[2]][i] = Averages[[2]][i]/countOld
67 }
68 colnames(Averages) <- c("Young", "Old")
69 rownames(Averages) <- rownames(df)
70 #write.csv(Averages, "Averages_Comparison.csv", row.names = TRUE)
71
72 #filtrado de aminoacidos con el valor de significancia
73 different_aminoacids <- subset(pvalues, Pvalues<0.05) #significance value

```

Figura 30. Código base para obtener el promedio de cada grupo y realizar filtrado por p values

Se realizan gráficas para ver de manera visual las diferencias entre las distribuciones de ambos grupos. La siguiente figura muestra histogramas y diagramas de caja y bigote. En donde se observa que el grupo de old es mucho mayor en cuanto a tamaños de muestra. Por otro lado se puede observar una pequeña diferencia visual entre ambos histogramas.

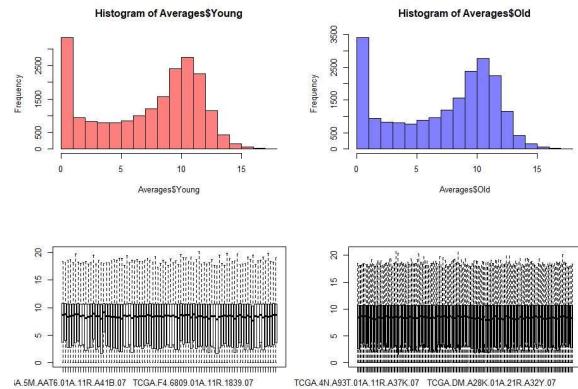


Figura 31. Histogramas y gráficas de caja y bigote para grupos estudiados

Con el fin de mostrar con mayor detalle la diferencia en cuanto a distribuciones fue necesario realizar una superposición de muestras en un histograma. Mostrando que en ciertos niveles de promedio en los genes

hay una diferencia significativa, en la que en algunos casos la muestra de adultos supera a la de jóvenes y viceversa.

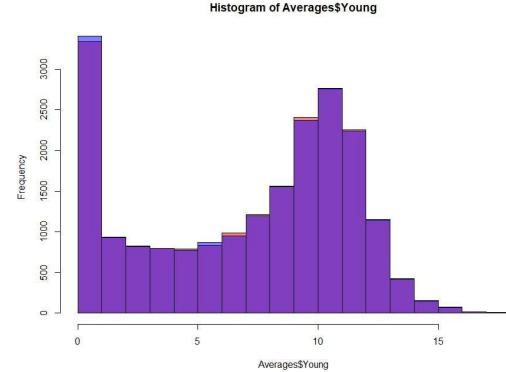


Figura 32. Superposición de histogramas de los grupos

A continuación se muestra un histograma con los genes ya filtrados a partir de los p values. Mostrando la tendencia general a encontrarse con diferencias del 0.02 en cuanto a p values. Estos genes pertenecen al grupo donde p value < 0.05.

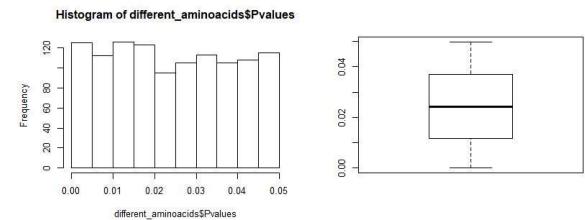


Figura 33. Histograma de p values y diagrama de caja y bigote

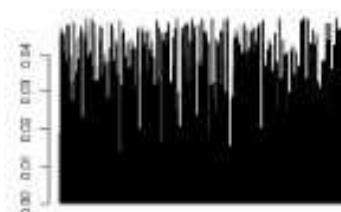


Figura 34. Gráfica de barras para cada gen a partir de su p value

Tras realizar este análisis se realiza un script para obtener un scatter plot de todos los p values y observar relaciones entre ellos. Para realizar esto se hace una prueba hipergeométrica con los genes diferencialmente expresados . Esto se hace ya que este tipo de prueba permite conocer a partir de muestreros aleatorios permitiendo obtener una función de masa probabilística permitiendo comprobar o descartar la hipótesis nula. Como primer paso se hace una lectura de los valores del t test de manera ordenada y de los valores obtenidos de la prueba hipergeométrica calculada a partir de la base de datos de *UC San Diego* y *Broad institute*. Esto con el fin de realizar un cómputo de superposiciones para los genes seleccionados. En este caso se obtuvieron 10 overlaps principales. En una colección de 25724 genes.

Collections	# Overlaps Shown	# Gene Sets in Collections	# Genes in Comparison (n)	# Genes in Universe (N)
C1, C2, C3, C4, C5, C6, C7, H	10	25724	1105	38404

Figura 35. Resultados de los overlaps encontrados en los genes diferencialmente expresados en t-test y prueba hipergeométrica

```

1 tTest <- read.csv("order_t_test.csv")
2 hyper <- read.csv("order_hipergeometric.csv")
3 namesTest <- matrix(tTest[,1])
4 namesHyper <- matrix(hyper[,1])
5 names <- rep(NA, times = (length(namesTest) + length(namesHyper)))
6
7 for(i in 1:length(namesTest)){
8   names[i] <- namesTest[i,1]
9 }
10 count <- length(namesTest)
11 for(i in 1:length(namesHyper)){
12   if (!namesHyper[i,1] %in% names){
13     count <- count + 1
14   names[count] <- namesHyper[i,1]
15 }
16 }
17 names <- sort(names)

```

Figura 36. Código base para lectura de base de datos y declaración de variables para grupos

Una vez importados se hace una iteración para encontrar aquellos valores que estén presentes en ambas pruebas estadísticas

permitiendo así crear una gráfica de dispersión.

```

19 vTest <- rep(NA, times = length(names))
20 vHyper <- rep(NA, times = length(names))
21
22 for (i in 1:length(names)) {
23   if (names[i] %in% namesTest){
24     vTest[i] <- tTest[match(names[i],namesTest),2]
25   }
26   if (names[i] %in% namesHyper){
27     vHyper[i] <- hyper[match(names[i],namesHyper),2]
28   }
29 }
30
31 scatter.smooth(vTest, vHyper)
32

```

Figura 37. Código para obtener gráfica de dispersión

Posteriormente para poder encontrar genes que realmente representen una diferencia estadística entre ambas muestras se hace una lectura de todos los tipos de genes de ambos conjuntos, es decir que se hace una búsqueda de aquellos genes que aparezcan tanto en la prueba hipergeométrica como en el t test para así poder analizar de manera formal como atacar este tipo de cáncer. Se hace una iteración para obtener todos los nombres comunes de los genes y guardarlos.

```

34 namesCommon <- rep("", times = length(namesTest))
35 count2 <- 0
36 for (i in 1:length(namesTest)){
37   if (namesTest[i,1] %in% namesHyper){
38     count2 <- count2 + 1
39     namesCommon[count2] <- namesTest[i,1]
40   }
41 }
42 namesCommon <- namesCommon[1:count2]
43
44 vTest2 <- rep(NA, times = length(namesCommon))
45 vHyper2 <- rep(NA, times = length(namesCommon))
46
47 for (i in 1:length(namesCommon)) {
48   vTest2[i] <- tTest[match(namesCommon[i],namesTest),2]
49   vHyper2[i] <- hyper[match(namesCommon[i],namesHyper),2]
50 }

```

Figura 38. Código para obtener genes comunes en prueba hipergeométrica y t-test

A continuación se hacen distintas iteraciones para obtener las posiciones y calcular diferencias en los p values.

```

54 vCommon = rep(NA, times = 10)
55 vSuma = rep(200, times = 10)
56 vPositionsTest = rep(0, times = 10)
57 vPositionsHyper = rep(0, times = 10)
58 count <- 0
59 pos <- 1
60
61 ~ for (i in 1:length(namesCommon)) {
62   suma <- vTest2[i] + vHyper2[i]
63   ~ if (suma < vSuma[10]){
64     j <- 1
65   ~ while (suma > vSuma[j] & j < 10){
66     j <- j + 1
67   }
68   n <- length(vSuma)
69   ~ while (n > j){
70     vPositionsTest[n] <- vPositionsTest[n-1]
71     vPositionsHyper[n] <- vPositionsHyper[n-1]
72     vSuma[n] <- vsuma[n-1]
73     n <- n - 1
74   }
75   vPositionsTest[j] = vTest2[i]
76   vPositionsHyper[j] = vHyper2[i]
77   vSuma[j] = suma
78 }
79 }
80

```

Figura 39. Código para obtener los genes comúnmente expresados y la diferencia en p values

Finalmente se guardan los primeros diez elementos ordenados de mayor a menor para genes comunes junto con sus p values y diferencias a partir de cada prueba hipergeométrica.

```

81 for (i in 1:10){
82   vCommon[i] = namesTest[match(vPositionsTest[i],tTest$Pos)]
83 }
84
85 table = data.frame(vCommon, vPositionsTest, vPositionsHyper, vSuma)
86 print(table)

```

Figura 40. Código base para mostrar los genes más diferencia de expresión

A partir de esto se obtuvieron los genes más significativos en ambas pruebas permitiendo así analizarlos y comprender

Gene Set Name	#Genes in Gene Set (K)	#Genes in Overlap (k)	k/ K	p-value	FDR q-value
MEISSNER_BRAIN_HCP_WITH_H3K4ME3_AND_H3K27ME3	1073	95	0.0885	3.34E-22	8.59E-18
BENPORAT_H_EED_TARGETS	1057	93	0.088	1.45E-21	1.86E-17
HALLMARK_TNFA_SIGNATING_VIA_NFKB	200	39	0.195	2.51E-21	2.15E-17

BENPORATH_ES_WITH_H3K27ME3	1112	94	0.0845	1.36E-20	7.3E-17
GO_TRANSCRIPTION_RELATION_REGULATOR_ACTIVITY	1718	123	0.0716	1.42E-20	7.3E-17
GO_DNA_BINDING_TRANSLATIONAL_ANSCRIPTON_FACTORIZATION_ACTIVITIY	1334	103	0.0772	1.28E-19	5.48E-16
AMIT_EGF_RESPONSE_40_HELA	42	18	0.4286	2.94E-17	1.08E-13
BENPORATH_SUZ12_TARGETS	1032	83	0.0804	5.15E-17	1.54E-13
GO_DNA_BINDING_TRANSLATIONAL_ANSCRIPTON_FACTORIZATION_ACTIVITIY_RNA_POLYMERASE_II_SPECIFIC	1094	86	0.0786	5.37E-17	1.54E-13
GO_POSITIVE_REGULATION_OF_RNA_METABOLIC_PROCESS	1710	114	0.0667	7.88E-17	2.03E-13

Figura 41. Tabla con los sets de genes con mayores overlaps y diferencia en expresión diferencial entre pruebas

A partir de estos resultados es posible realizar una búsqueda en DGIdb para conocer drogas que pueden ser efectivas para el tratamiento directo de los genes diferencialmente expresados. Únicamente fueron encontrados cuatro genes que pueden tener efectos interactivos con drogas.

The figure consists of four separate search results from a database:

- GJB2:** Shows interactions with CARBENOXOLONE, c2A, FLUENAMIC ACID, and OCTANOL, all listed as inhibitors.
- ATF3:** Shows interactions with PROGESTERONE, listed as n/a.
- LRP2:** Shows interactions with UROKINASE, INSULIN HUMAN, and CARBOQUONE, all listed as n/a.
- NEUROD1:** Shows interactions with DEFEROXAMINE and CHEMBL35482, both listed as n/a.

Figura 42. Medicamentos e interacciones efectivas con principales genes

8. Clasificador de cáncer colorrectal

Con el fin de comprender con mayor exactitud el funcionamiento del cáncer colorrectal se realizó un algoritmo para poder determinar clusters. En este caso se aplicó técnicas de unsupervised learning. La obtención de grupos se realizó con k-means, este algoritmo busca encontrar grupos al minimizar la distancia entre las observaciones buscando óptimos locales como solución. La distancia es obtenida a partir de coordenadas de observación en la matriz. Como primer paso se aplican posiciones aleatorias minimizando las distancias entre los puntos buscando encontrar centroides para cada grupo. Esto se realiza de manera iterativa hasta no encontrar cambios entre los grupos.

$$distance(x, y) = \sum_i^n (x_i - y_i)^2$$

El valor de k se definió a partir de los dos grupos del dataset, a partir de esto se aplica el algoritmo obteniendo un método para

clasificar a partir de expresión genética, permitiendo encontrar si un paciente padece o no de cáncer. Para esto se importa la base de datos de TCGA con información de distintos pacientes. Posteriormente se realiza el algoritmo para dos clusters.

```

70 #Clustering young
71 set.seed(2340)
72 df.young <- kmeans(df, 2)
73 kmeans.analyse(df, 2)
74 df_cluster <- kmeans(df, 2)
75
76 kmean_withinss <- function(k) {
77   cluster <- kmeans(df, k)
78   return (cluster$tot.withinss)
79 }
80 print("Total within clusters sum of squares")
81 print(kmean_withinss(2))
82
83 #start of algorithm
84 # Set maximum cluster
85 max_k <- 20
86 # Run algorithm over a range of k
87 wss <- sapply(2:max_k, kmean_withinss)
88 elbow <- data.frame(2:max_k, wss)
89 # Plot the graph with elbow
90 gplot(elbow, aes(x = X2.max_k, y = wss)) +
91   geom_point() +
92   geom_line() +
93   scale_x_continuous(breaks = seq(1, 20, by = 1))
94
95 print("Cluster size")
96 print(young_cluster$size)

```

Figura 43. Código para obtención de k clusters

A continuación se muestran un grupo de gráficas en la que se muestra la evolución de un punto aleatorio hacia los centros de los grupos permitiendo así después de varias iteraciones obtener los dos centroides y por consiguiente los grupos de old y young a partir de los genes.

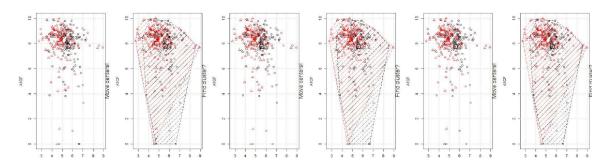


Figura 44. Evolución de clusters generales

En la siguiente figura se observa el resultado del algoritmo, el cual clasifica los datos en dos clusters. Los cuales son útiles para poder detectar a partir del código genético del tumor si este es cancerígeno o no.

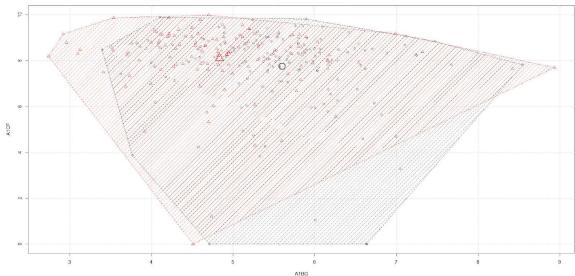


Figura 45. *K* clusters (*jóvenes* y *adultos*)

Una de las ventajas del análisis de expresión diferencial es que con ella se pudo obtener información sobre los distintos tipos de mutaciones. Por lo que se optó por crear sub clusters para cada grupo estudiado. Esto con el fin de poder identificar condiciones como la de HNPCC, permitiendo así clasificar con mayor exactitud las ubicaciones de las mutaciones y tratarlas de la manera más efectiva posible. Para estos dataframe se decidió debido a la ambigüedad de mutaciones significativas utilizar un algoritmo para obtención óptima del valor *k*. En este caso se utilizó el método “elbow”. El cual utiliza la homogeneidad dentro del grupo para evaluar la variabilidad.

Trabajando así con el porcentaje de la varianza expresada en cada grupo. A medida que los grupos crecen la variabilidad aumenta mientras la heterogeneidad disminuye. Este algoritmo permite encontrar el valor de *k* que está más allá de los rendimientos decrecientes. Permitiendo encontrar este punto a partir de la suma total de cuadrados de los clusters.

Como primer paso se obtienen los valores a partir de la suma, los cuales son probados con los valores del *k* preestablecido, corriendo el algoritmo sobre el rango de *k* siendo así más eficiente que haciéndolo de manera iterativa. Esto hasta encontrar el

punto de inflexión en la curva. Esta función es graficada permitiendo obtener el valor adecuado para el algoritmo de agrupamiento.

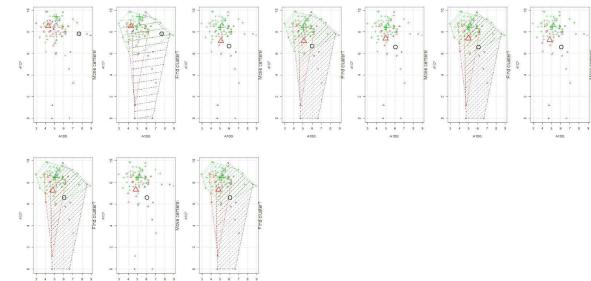


Figura 46. Evolución de clusters en jóvenes

En estas gráficas es posible mostrar la evolución del algoritmo a partir del *k* idóneo, el cual se situó en 3. Mostrando así tres mutaciones significativas en el grupo de jóvenes dentro de las cuales se puede encontrar el síndrome de Lynch.

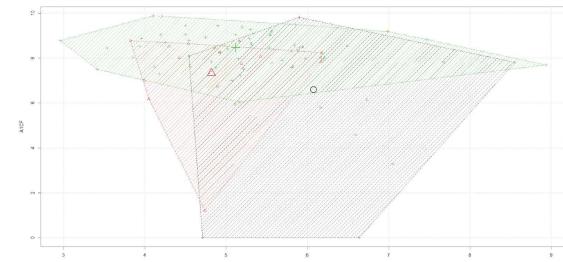


Figura 47. *K* clusters en jóvenes

De igual manera se reproduce el valor de *k* igual a 3 para el dataframe de *old*, indicando así 3 mutaciones principales para ambos grupos. Dentro de los cuales uno de los clusters es atribuido a la mutación hereditaria.

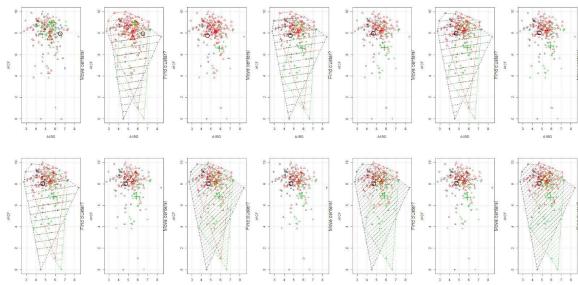


Figura 48. Evolución de clusters en adultos

Finalmente se encuentra el último grupo de clusters para el conjunto old. Estos clusters demostraron ser altamente efectivos con un tamaño común entre old y young de 46,24 y 14 como tamaño. Este modelo podría ser utilizado como método para análisis de tumores, permitiendo conocer a qué tipo de mutación pertenecen, características genéticas por edad y a partir del tipo de mutación el tratamiento médico más efectivo.

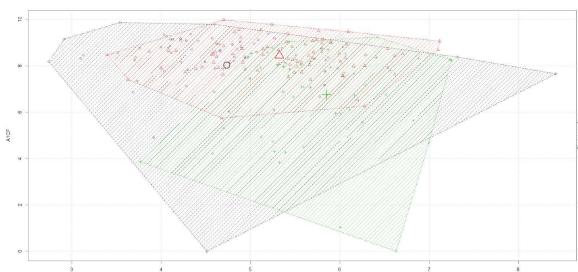


Figura 49. K clusters en adultos

9. Conclusiones

El aumento de los casos de cáncer colorrectal se ha vuelto una preocupación muy grande para la comunidad médica, especialmente porque el CCR se presenta cada vez más en adultos jóvenes. La temprana detección de esta enfermedad puede ser la diferencia entre la vida y la muerte de un paciente. Es por esto que cada vez se están desarrollando más tratamientos para combatir esta enfermedad.

El análisis de mutaciones de genes es una gran forma de lograr una detección temprana de esta enfermedad.

Decidimos un método de cluster con machine learning para la detección de cancer en los genes. Lo que muestra esta gráfica es la agrupación de mutaciones que se detectan en el genoma ingresado. El número de clusters utilizado fue de 3 debido a lo encontrado con el metodo de optimizacion “elbow”. También es una limitación de procesamiento que tenemos por la inmensa cantidad de datos a analizar.

10. Trabajo futuro

Esta investigación ha permitido explorar distintas disyuntivas del cáncer colorrectal en la población mexicana por lo que a futuro el modelo de clustering debería de ser probando en conjuntos de datos más grandes y variados. Esto con el fin de encontrar sesgos, reducirlos y mejorar el modelo. Además de esto realizar el análisis de expresión diferencial podría ser más significativo en un grupo de datos mucho más variado y vasto. Finalmente a partir de este modelo de clustering y los datos genéticos se podrían desarrollar modelos para la medición de efectividad de tratamientos para cada grupo de cada sector afectado por el cáncer. Específicamente podría ser mejorado con el uso de redes neuronales recurrentes, las cuales mejoran la precisión del modelo y podrán realizar análisis genético con mayor eficacia.

Referencias

Bronner, C. E., Baker, S. M., Morrison, P. T., Warren, G., Smith, L. G., Lescoe, M. K., ... & Tannergård, P. (1994). Mutation in the DNA mismatch repair gene homologue hMLH 1 is associated with hereditary non-polyposis colon cancer. *Nature*, 368(6468), 258-261.

Das, S., Ciombor, K.K., Haraldsdottir, S. et al. Promising New Agents for Colorectal Cancer. *Curr. Treat. Options in Oncol.* 19, 29 (2018).

Ganesh, K., Stadler, Z., Cerek, A., Mendelsohn, R., Shia, J., Segal, N., & Díaz, L. (2019). Immunotherapy in colorectal cancer: rationale, challenges and potential. *Nature Reviews Gastroenterology & Hepatology*, 361–375 .

Kelsy C Cotto, Alex H Wagner, Yang-Yang Feng, Susanna Kiwala, Adam C Coffman, Gregory Spies, Alex Wollam, Nicholas C Spies, Obi L Griffith, Malachi Griffith, DGIdb 3.0: a redesign and expansion of the drug–gene interaction database, *Nucleic Acids Research*, Volume 46, Issue D1, 4 January 2018, Pages D1068–D1073, <https://doi.org/10.1093/nar/gkx1143>

Liu Fangqi, Cai Sanjun. Perioperative adjuvant and neoadjuvant therapy for colorectal cancer [J]. *Chinese Journal of Gastrointestinal Surgery*, 2019,22 (4): 315-320. DOI: 10.3760 / cma.j.issn. 1671-0274.2019.04.002

Lin, K.M., Shashidharan, M., Ternent, C.A. et al. Colorectal and extracolonic cancer variations in MLH1/MSH2 hereditary nonpolyposis colorectal cancer kindreds and the general population. *Dis Colon Rectum* 41, 428–433 (1998).
<https://doi.org/10.1007/BF02235755>

Lynch, H. T., & Smyrk, T. (1996). Hereditary nonpolyposis colorectal cancer (Lynch syndrome): an updated review. *Cancer: Interdisciplinary International Journal of the American Cancer Society*, 78(6), 1149-1167.
 MLH1 - Gene - NCBI. (2020). Retrieved 29 April 2020, from <https://www.ncbi.nlm.nih.gov/gene/4292>

MSH2 - Gene - NCBI. (2020). Retrieved 29 April 2020, from <https://www.ncbi.nlm.nih.gov/gene/4436>

Sankila, R. I. S. T. O., Aaltonen, L. A., Jarvinen, H. J., & Mecklin, J. P. (1996). Better survival rates in patients with MLH1-associated hereditary colorectal cancer. *Gastroenterology*, 110(3), 682-687.

K-means Clustering in R with Example. (2020). Retrieved 4 May 2020, from <https://www.guru99.com/r-k-means-clustering.html>

Código utilizado en el proyecto:
https://github.com/SeaWar741/ITC/tree/master/2do_Semestre/Biologia_computacional/Final