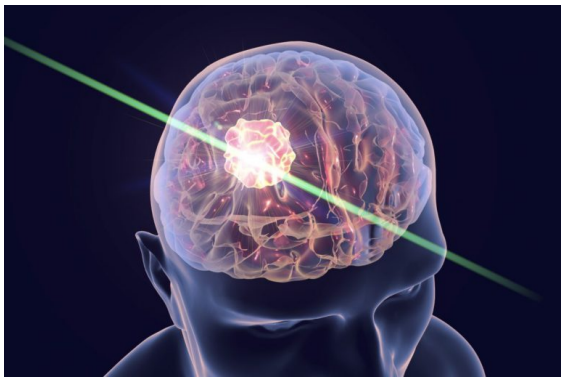
Memoria AA

Detección de tumores



Xabier Jiménez Gómez
Daniel García Paz
Saúl Leyva Santarén
Diego Fresco Rivera

Índice

1. Introducción

Los tumores cerebrales son considerados actualmente como una de las enfermedades más graves y difíciles de tratar en todo tipo de edad, tantos niños como adultos. Dentro de la categoría de tumor, los cerebrales representan la gran mayoría de los mismos del Sistema Nervioso Central. Dichos tumores que cuentan con una considerable tasa de mortalidad se clasifican en:tumor benigno, tumor maligno, tumor hipofisario etc. Además para su correcto tratamiento es imprescindible su detección temprana mediante resonancias magnéticas lo cual genera grandes cantidades de datos por imágenes de los escaneos.

La IA y el aprendizaje automático demuestran un aumento en la precisión para la clasificación de los mismos y su detección en los órganos afectados con respecto a una clasificación manual por parte de trabajadores. Por tanto se toma la decisión de realizar un sistema de detección de tumores cerebrales mediante imágenes utilizando una serie de algoritmos y técnicas de Deep Learning.

Ventajas de resolver esta problemática:

1. Cada tumor puede tener un tamaño y forma distinta por lo tanto en ocasiones puede ser difícil detectar el mismo a simple vista y resulta importante resolver esta complejidad con gran precisión.
2. Se trata de un tema de alta importancia y riesgo y por tanto cuantos mas estudios y avances haya sobre el tema la documentación cada vez será mas rigurosa y en aumento.

Desventajas de no resolver esta problemática:

1. El no avanzar en el estudio de estos temas cruciales del sector sanitario conlleva numerosos problemas en el futuro que podrían ser previstos.

Este problema ha sido tratado en muchas ocasiones y con sistemas mucho más sofisticados que el que se va a proponer, ya que se tratará de una implementación sencilla y concreta para su clasificación en existencia de tumor o su inexistencia.

El aprendizaje automático es el subcampo de las ciencias de la computación y una rama de la inteligencia artificial, cuyo objetivo es desarrollar técnicas que permitan que las computadoras aprendan. En las sucesivas aproximaciones trataremos de solucionar este problema utilizando distintas herramientas de aprendizaje automático. En cada una de ellas trataremos de obtener una solución por medio del uso de RR.NN.AA, SVMs, árboles de

decisión y el algoritmo KNN, modificando algunos de sus parametros de entrada como se detalla más adelante.

Las RR.NN.AA consisten en un conjunto de unidades llamadas neuronas, conectadas entre sí para transmitirse señales. De esta forma se simula el funcionamiento de un cerebro biológico.

El algoritmo SVM tiene como objetivo encontrar un hiperplano que separe, de la mejor forma posible, dos tipos diferentes de datos.

Un árbol de decisión es una estructura similar a un diagrama de flujo donde un nodo interno representa una característica (o atributo), la rama representa una regla de decisión y cada nodo hoja representa el resultado.

KNN es un algoritmo basado en instancia de tipo supervisado. Sirve para clasificar valores buscando los puntos de datos más similares (por cercanía) aprendidos en la etapa de entrenamiento y haciendo conjeturas de nuevos puntos basado en esa clasificación.

La memoria constará de introducción, descripción general del problema (en el que se aporta información sobre la Base de Datos y otros aspectos), análisis bibliográfico (donde se referencian artículos de proyectos relacionados, de los que se ha extraído información útil) y por último, desarrollo.

2. Descripción del problema

Nuestro cometido es contruir un sistema de aprendizaje que, a partir de un conjunto de muestras, analice y determine si el tumor está presente. Por ahora tan solo nos centramos en su detección, posteriormente se podría hacer una clasificación según el tipo[1].

2.1. Restricciones

- Las imágenes seleccionadas para la base de datos se encuentran en escala de grises, es decir, posee un valor de graduación de gris. En el caso de que se usen imágenes a color no tendría efecto, ya que la media y desviación típica de las imágenes se sacan con la escala de grises y no con RGB.

- Solo podrá haber un tumor por imagen. En la base de datos solo hay imágenes con un tumor. Esto hace que a la hora de recortar las imágenes el tumor se encuentra ajustado al marco de la misma y por este motivo solo coge un tumor por imagen.

2.2. Propiedades de los datos

Para esta práctica se utilizará una base de datos de imágenes obtenida de Kaggle (Figura 2). Se han cogido un número arbitrario de imágenes y han sido recortadas, dado que hacerlo con una imagen completa es un problema muy complejo, comenzaremos analizando imágenes mucho más pequeñas que las originales. Al recorte no se le ha aplicado ningún criterio ya que los tamaños del tumor pueden variar respecto a la posición de la imagen tomada. Nuestra base de datos, actualmente, cuenta con 120 imágenes divididas en dos carpetas, 60 con tumor y 60 sin tumor.

Además, para valorar y comparar los clasificadores que se generan utilizamos como métricas la media y la desviación típica de los colores(escala de grises) ya que contamos con tumores monocromáticos que pueden tener el valor blanco o negro únicamente. Cabe destacar que obviamos la existencia de falsos positivos en las comprobaciones de las imágenes y nos centramos en que el algoritmo siempre detecte la existencia del tumor,por tanto en cuánto a sensibilidad se refiere, siempre tenemos que detectarlo aunque en ocasiones demos un caso inexistente ya que lo grave sería no detectarlo y que avance sin su conocimiento. Y finalmente por otro lado es obviada la existencia de falsos negativos pero se el objetivo es disminuírlos el máximo posible ya que la totalidad de los verdaderos negativos nos daría un sistema con funcionamiento correcto.

[1]Las muestras han sido reducidas a imágenes de menor tamaño para reducir su complejidad

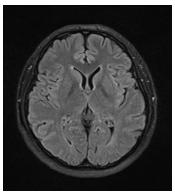


Imagen sin tumor

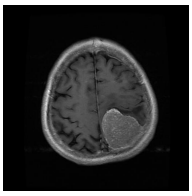


Imagen con meningioma

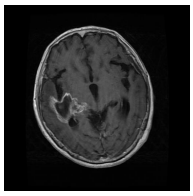


Imagen con glioma

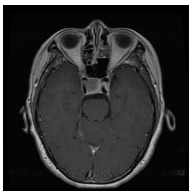


Imagen con tumor pituitario

Figura 2: Imágenes base de datos

3. Análisis bibliográfico

Para este apartado de la memoria se estudiarán casos de estudios los cuales tengan cierta relación con el tema a tratar y hayan servido como inspiración u/o ayuda para la elaboración del mismo.Por ende se enumeran y describen diferentes trabajos ya terminados con resultados públicos.

En el estudio realizado por el organismo de neurocirugía de la Universidad de Michigan [4] se quiso comprobar que los algoritmos creados identificaban patrones analizando las imágenes y así extraer carácterísticas de los mismos.A su vez en la Universidad de Pennsylvania,tanto Intel Labs como la Facultad de Medicina de Perelman están colaborando en el entrenamiento de modelos de inteligencia artificial (IA) para identificar tumores cerebrales mediante el aprendizaje federado(técnica de aprendizaje automático que entrena un algoritmo a través de una arquitectura descentralizada formada por múltiples dispositivos los cuales contienen sus propios datos locales y privados).Mientras,la famosa empresa Google está comprometida en la creación de una IA para la detección de cáncer de mama y de pulmón con una reducción importante de falsos positivos abriendo así un nuevo camino en la investigación de estas tecnologías diagnósticas utilizando el framework DeepPATH [6] estudiando el aprendizaje profundo.

Por otro lado,en publicaciones de kaggle y periódicos de ciencia como Arabian Journal for Science and Engineering,encontramos numerosos proyectos de código abierto con bases de datos muy similares a la nuestra.Trabajos como el de Dbeiver y Ewerton Santos [8] o Sajid [7] utilizan gran cantidad de imágenes con escalas a dos colores (como en nuestro caso) para clasificar las mismas mediante entrenamiento neuronal con una serie de características de primer orden.Estas son media,varianza,desviación,asimetría y curtosis entre otras y por otro lado existen características de segundo orden como entropía,contraste y homogeneidad.

Sin ir más lejos encontramos el proyecto de detección de tumores realizado por Jayachandran [5] entre otros, del cual hemos basado gran parte de nuestro trabajo hasta el momento.Para clasificar tumores utilizan Deep Learning y examinan la posición del tumor y lo clasifican en glioma,meningioma y pituaitary a diferencia de nuestro proyecto que por el momento sólo detecta la inexistencia o existencia del mismo por el momento.

Otros ejemplos de trabajos de campo muy similares [9] detectan canceres de vejiga a partir de fotogramas de video o el proyecto de Ali Ari y Davut Hanbay [2] realizado en Turquía a partir de tres etapas;preprocesamiento,clasificación tumoral basada en campos receptivos locales de la máquina de aprendizaje extremo y la extracción de la región tumoral por medio de imágenes.Por último,otros proyectos más sofisticados serían el creado por Hao Dong, Guang Yang, Fangde Liu, Yuanhan Mo y Yike Guo [3] que evalúa hasta el grado del tumor en alto o bajo y el estudiado por Javeria Amin, Muhammad Sharif, Mussarat Yasmin

y Steven Lawrence Fernandes [1] en el cuál también se evalúa su grado con validaciones en los resultados de precisiones complicadas como el coeficiente de similitud de datos o similitud de Jaccard pero en el trabajo ha realizar no nos meteremos en tanta profundidad.Estos dos últimos trabajos consiguieron resultados comparables con otros métodos de vanguardia.

Estos trabajos se utilizarán como fuente de inspiración y comparación con nuestro proyecto siempre y cuando tengan fines similares.Destacamos que las aproximaciones serán realizadas exclusivamente por nosotros.

4. Desarrollo

A continuación se mostrarán las distintas aproximaciones durante el desarrollo del sistema, hechas a partir de la descripción del problema. Para cada una de ellas habrá una descripción, en donde se explica como se ha llevado a cabo dicha aproximación; los resultados obtenidos y una discusión, en donde daremos el porqué de estos.

Utilizaremos la técnica de validación cruzada con 10 subconjuntos de datos (10-fold cross-validation) para evaluar los resultados y garantizar que son independientes de la partición entre datos de entrenamiento y test. Cada una de las combinaciones de subconjuntos de entrenamiento y test será entrenada un total de 75 veces y se utilizará la media de estos entrenamientos como datos finales de esa topología.

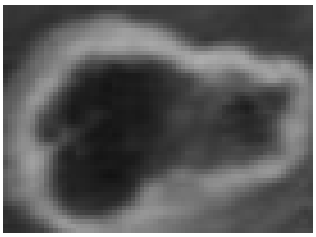
Como algoritmos de aprendizaje automático usaremos redes de neuronas artificiales (RR.NN.AA), máquinas de soporte vectorial (SVM), árboles de decisión y kNN.

Para valorar los algoritmos mediremos su sensibilidad y especificidad, anteponiendo la primera. El motivo para utilizar las 2 es que un sistema podría tener una sensibilidad enorme a costa de etiquetar todos los casos como positivos lo cual, obviamente, no proporciona un sistema válido.

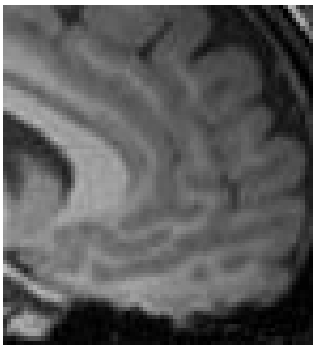
4.1. Primera aproximación

4.1.1. Descripción

Como primera toma de contacto decidimos que el sistema detectara si hay o no un tumor en la imagen. A partir del conjunto de imágenes de entrenamiento obtenidas a partir de TC, hemos recortado un conjunto de 120 imagenes de las cuales 60 presentaban tumor y 60 no lo presentaban. Cada una de ellas tiene una resolución entre 200x200 y 400x400 pixeles. Estas imágenes presentan gliomas y meningiomas facilmente reconocibles a simple vista, dejando a los tumores pituitarios fuera del dataset porque su detección visual es más compleja. Para las imágenes recortadas de los tumores ajustamos la forma del tumor a los límites de la imagen, como se puede ver en la Figura 3. En cuanto a las imágenes recortadas sin tumor, las sacamos tanto de imágenes sin tumor, como de imágenes con meningioma o glioma; pero mostrando una zona sana del cerebro sana, como la imagen de la Figura 3. Debido a que los tamaños de los tumores pueden ser muy distintos, la resolución de las imágenes recortadas también lo es, pudiendo oscilar entre 50x50 y 150x150 pixeles aproximadamente.



(a) Con tumor



(b) Sin tumor

Figura 3: Imagenes recortadas

Debido a la gama de colores de las muestras, optamos por usar una escala de grises para la extracción de las características. De esta forma el sistema diferenciará las imágenes por la media y desviación típica entre ellas. Cada uno de los valores estará normalizado entre 0 y 1. Los datos quedarían como en el Cuadro 1.

Patrones			
Row	Mean	Std	Target
1	0.321047	0.120569	1
2	0.295845	0.125879	1
...	...	...	...
90	0.264009	0.100171	0
91	0.469917	0.259031	0
...	...	...	...

Cuadro 1: Tabla de patrones

4.1.2. Resultados

Como ya mencionamos anteriormente, los atributos a utilizar en esta aproximación para la detección de tumores son la media y la desviación típica (normalizadas entre 0 y 1) de la escala de grises de las imágenes recortadas. Como se puede ver en la Figura 4, aunque se podría llegar a acotar un area aproximada donde el número de tumores es mucho mayor que el número de no tumores, la cantidad de ruido es excesiva. Esto, como veremos, generará un problema de cara a la efectividad de nuestras redes pues podemos observar que varios puntos de tumores y no tumores son practicamente coincidentes.

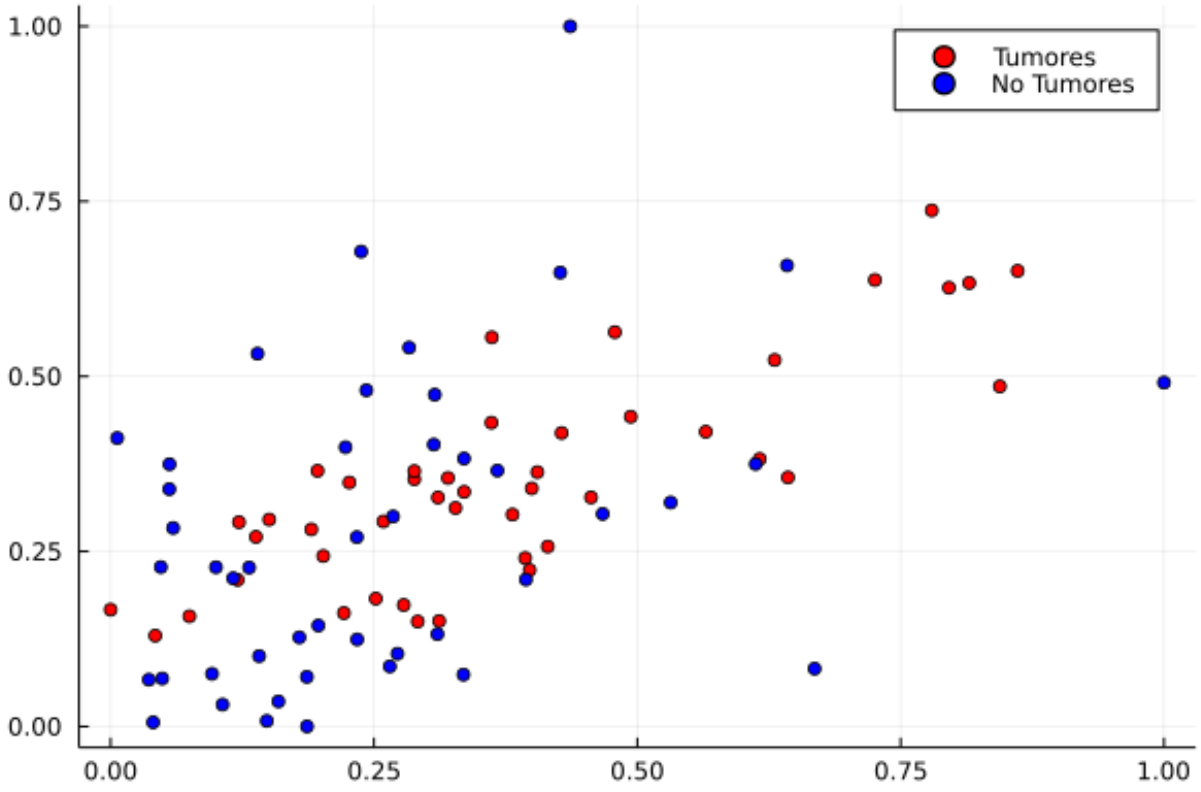


Figura 4: Distribución de los atributos del dataset

Para cada uno de los algoritmos, repetimos varias veces su entrenamiento y cogimos aquellos que tuvieran una mayor sensibilidad. En cuanto a las topologías, probamos con 1 y dos capas ocultas de la forma [1], [1,1], [1,2], ..., [10], [10,1], ..., [10,10].

- **RR.NN.AA**

  Como se ha mencionado anteriormente, se han probado un total de 110 topologias, desde la [1] hasta la [10,10]. Para el entrenamiento de cada una de estas topologias se han utilizado los siguientes parámetros de entrenamiento:

  1. Ratio de aprendizaje: 0.01
  2. Número máximo de ciclos de entrenamiento: 10000
  3. Número de ciclos sin mejora en la validación: 250
  4. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  5. Número de iteraciones por subconjunto: 75
  6. Ratio de elementos entrenamiento/validacion: 0.8/0.2

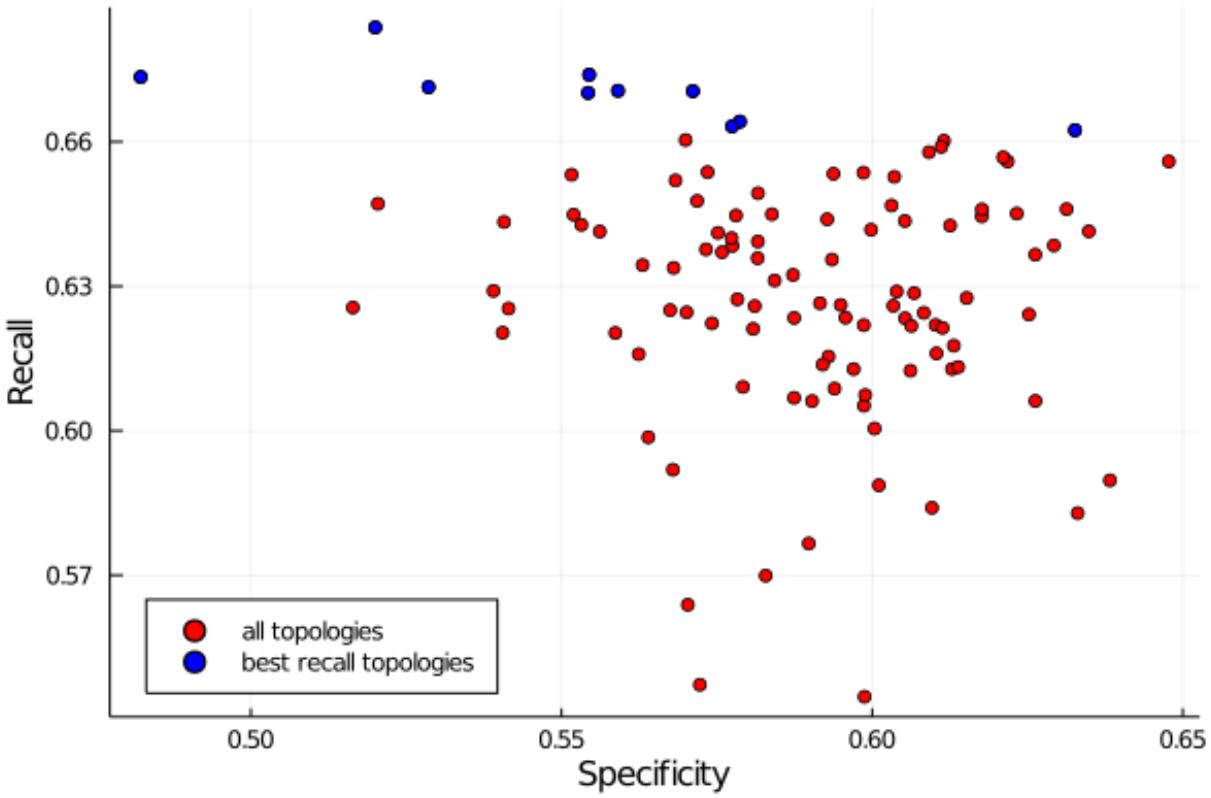


Figura 5: Valores de sensibilidad y especificidad obtenidos

Tras el entrenamiento obtenemos la siguiente distribución de resultados (Figura 8), de las cuales podemos determinar que las mejores son las topologías:

Mejores resultados		
Topología	Sensibilidad	Especificidad
[7]	0.6624	0.6326
[6,2]	0.6641	0.5787
[8,2]	0.6705	0.5711

Cuadro 2: Tabla de mejores resultados

Vale la pena resaltar que, a pesar de que nuestro sistema prioriza la sensibilidad por encima de la especificidad, tenemos esta última muy en cuenta, ya que si nos fijamos bien, la red neuronal que consideramos óptima es la mas específica dentro de las más sensibles. Esto lo hacemos porque como comentamos antes un sistema que de todos los resultados como positivos será muy sensible, pero será totalmente inútil.

Una vez seleccionada la topología con una capa oculta de 7 neuronas como la mejor para resolver nuestro problema podemos ver tambien su matriz de confusión obtenida a partir del cálculo de la media de verdaderos y falsos positivos y negativos de cada un de los distintos tests (Cuadro 3).

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	3,644	1,857	5,501
	Negativo	2,054	3,536	5,590
	Total	5,698	5,393	N

Cuadro 3: Matriz de confusión

Podemos ver que, como esperábamos, el número de aciertos es superior al de fallos tanto en la detección de positivos como en la de negativos, el problema reside en que esta diferencia es insuficiente, siendo ampliamente superada por un examen visual, incluso por el de un ojo inexperto. Podemos concluir pues que no hemos obtenido un modelo válido que resuelva nuestro problema satisfactoriamente.

- **SVM**

Como parámetros de entrenamiento de las SMV utilizamos:

1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
2. Cuatro tipos de kernel: lineal, polinomial, RBF y sigmoide
3. C: 0.1, 1, 10, 100, 1000, 10000
4. Degree: 3
5. Gamma: 2

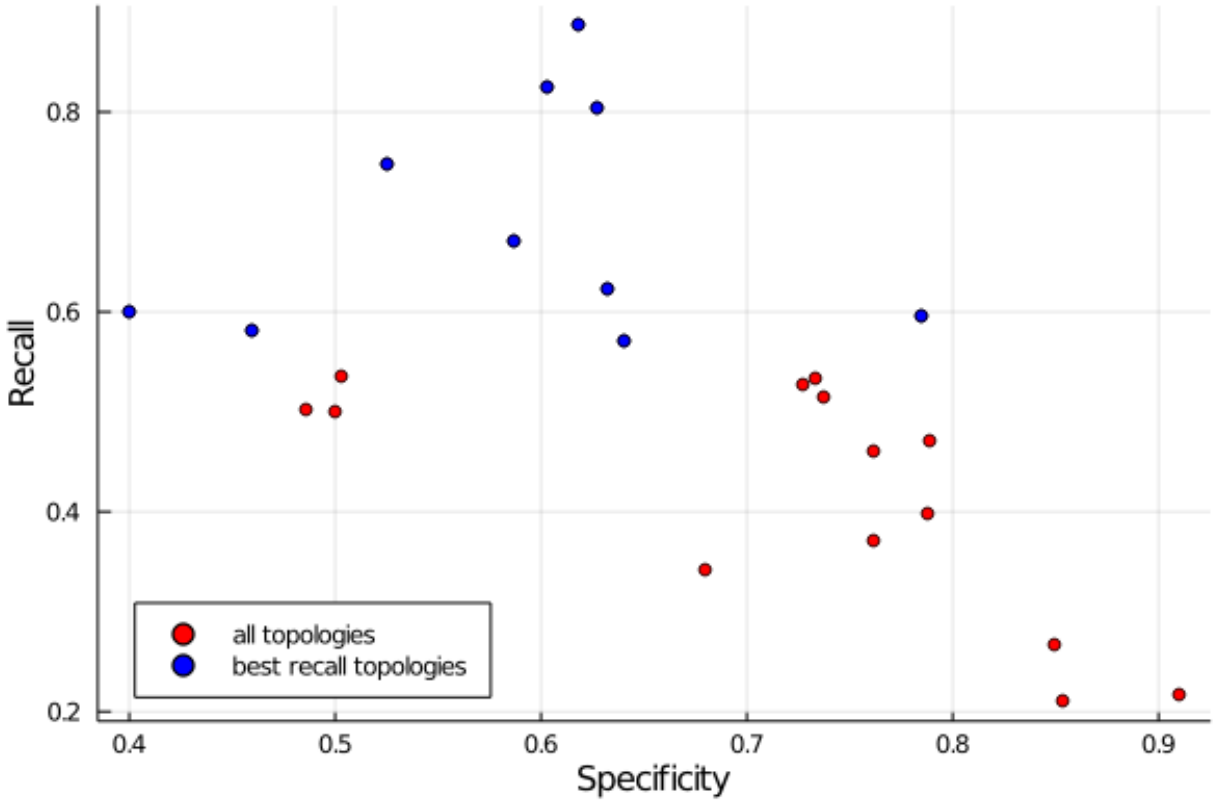


Figura 6: Valores de sensibilidad y especificidad obtenidos

Teniendo en cuenta que los mejores resultados son aquellos que están más arriba a la derecha, y que para nuestro problema nos interesa el porcentaje de sensibilidad, los mejores resultados son:

Mejores resultados		
Kernel y C	Sensibilidad	Especificidad
rbf-1000.0	0.8863	0.6180
rbf-10000.0	0.8247	0.6039
rbf-100.0	0.8023	0.6280
poly-10000.0	0.7459	0.525

Cuadro 4: Tabla de mejores resultados

Los tres mejores resultados fueron con el kernel RBF. El primero con C=1000, el segundo con C=10000 y el tercero tenían C=100. En cuarto lugar aparecería el kernel

polinomial con C=10000. Viendo estos resultados se puede concluír que se obtienen mejores resultados con el kernel RBF y valores altos de C.

Como se puede apreciar se ha conseguido una sensibilidad del 88 % y una especificidad del 61 %. Tanto el segundo como el tercer mejor resultado tienen valores similares: 80 % de sensibilidad y 60 % de especificidad.

La matriz de confusión para el mejor resultado (rbf1000.0) sería:

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,5	0,8	5,3
	Negativo	2,0	3,3	5,3
	Total	6,5	4,1	N

Cuadro 5: Matriz de confusión

Como podemos ver en el Cuadro 5, el número de verdaderos positivos es superior al de falsos positivos y muy superior al de falsos negativos. Además el número de verdaderos negativos es elevado, por lo que la clasificación no ha sido mala.

En nuestro problema nos interesa una alta sensibilidad, para lo cual podriamos considerar que este algoritmo es bastante competente, no obstante la especificidad si que es un bastante baja, por lo que deberiamos intentar mejorarla en futuras aproximaciones.

- **Árboles de decisión**

En cuanto a los parámetros de entrenamiento de los árboles de decisión son:

1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
2. Max Depth: de 1 a 20
3. Random State: 1

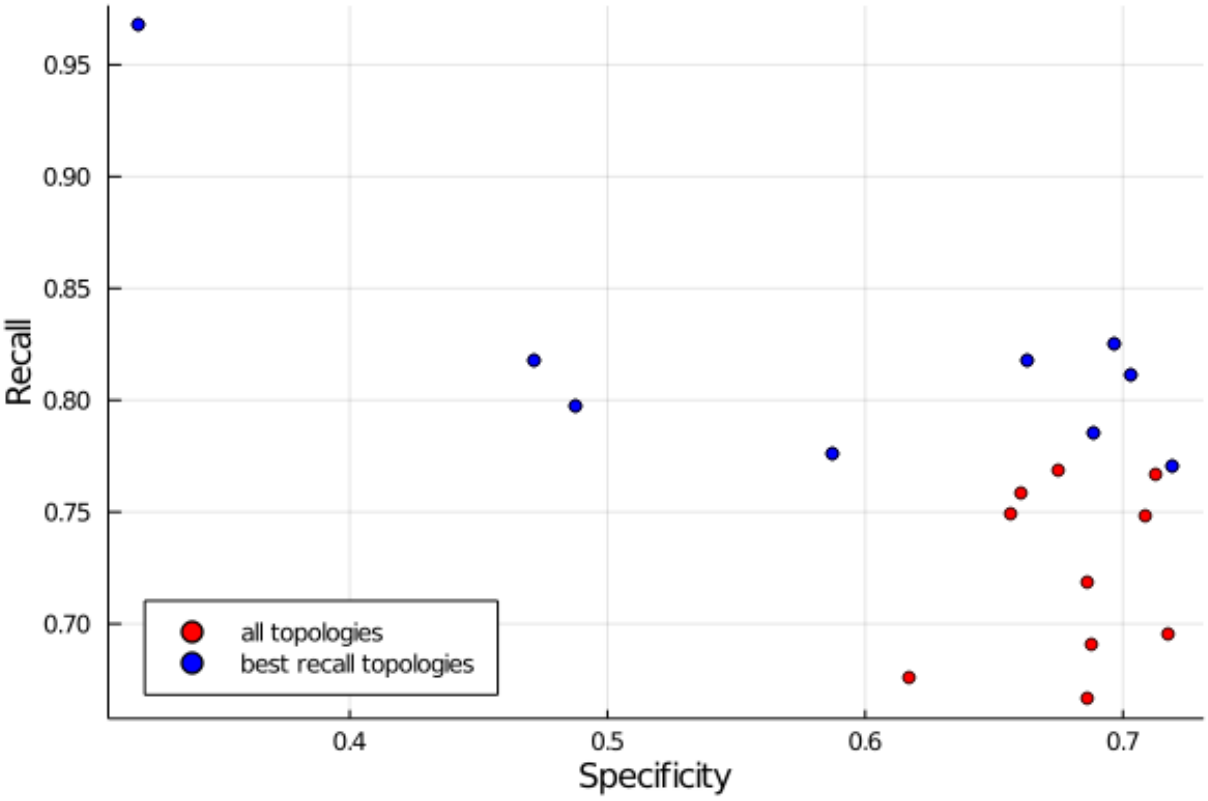


Figura 7: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Profundidad	Sensibilidad	Especificidad
7	0.8248	0.6966
8	0.8111	0.7038
20	0.8171	0.6635

Cuadro 6: Tabla de mejores resultados

Los resultados no son similares como podían serlo en las SVM. Hay resultados con una alta sensibilidad pero muy baja especificidad, y resultados con sensibilidad y especificidad media. El mejor resultado entre todos ellos es el de profundidad 7, ya que tiene una sensibilidad del 82 % y una especificidad del 69 %. Le seguirían el de profundidad 8 (81 % de sensibilidad y una especificidad del 70 %) y profundidad 20 (81 % de sensibilidad y una especificidad del 66 %). Aunque la especificidad es mayor que en las SVM, para nuestro problema estos resultados son peores que los de SVM. La matriz de confusión del mejor resultado sería la siguiente:

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,4	0,9	5,3
	Negativo	1,8	3,5	5,3
	Total	6,2	4,4	N

Cuadro 7: Matriz de confusión

Como se puede observar en el Cuadro 7, los verdaderos positivos y negativos son elevados mientras que los falsos negativos son bajos. Sin embargo, los falsos positivos no son todo lo bajos que nos gustaría. Podemos concluir por tanto que este sistema es lo suficientemente sensible como para dar un resultado relativamente satisfactorio, pero aun así debería mejorarse en futuras aproximaciones.

- **kNN**

  Los parámetros utilizados para el algoritmo kNN son:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Número de neighbors: de 1 a 20

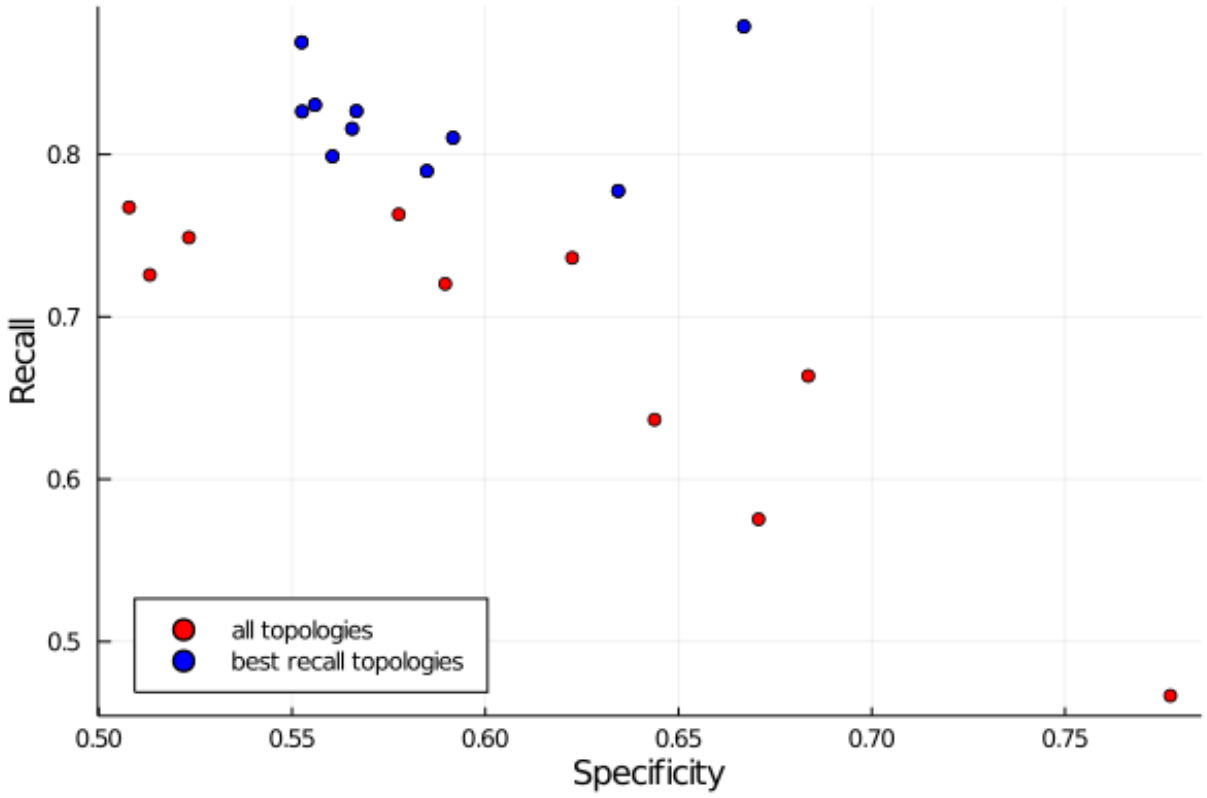


Figura 8: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Vecinos	Sensibilidad	Especificidad
3	0.8788	0.6669
17	0.8690	0.5525
20	0.8304	0.5559

Cuadro 8: Tabla de mejores resultados

Podemos apreciar que conseguimos valores altos, rozando el 90 % en sensibilidad y resultados asequibles en especificidad.

También destacamos otros resultados que se acercan al óptimo estudiado anteriormente. Entre estos destacamos el entrenado con 17 vecinos dando como resultados una sensibilidad de 86.90 % y especificidad de 55.25 % dando por tanto valores practicamente iguales al detectar positivos que es lo más importante en nuestro estudio pero con una tasa de dectección de negativos considerablemente mas baja. Posteriormente con 20 vecinos encontramos resultados de 83.04 % y 55.59 % en sensibiliad y especificidad respectivamente.

Después de su respectivo entrenamiento se ha detectado como la mejor métrica la obtenida con 3 vecinos dando los siguientes resultados:

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	$4,6$	$0,7$	5,3
	Negativo	$1,9$	$3,4$	5,3
	Total	6,5	4,1	N

Cuadro 9: Matriz de confusión

Como podemos ver en el Cuadro 9, el número de verdaderos positivos es superior al de falsos positivos y muy superior al de falsos negativos. Además el número de verdaderos negativos es elevado, por lo que la clasificación ha cumplido las expectivas que daban los anteriores algoritmos,sin embargo la tasa de falsos positivos aun podria ser reducible. Una vez mas podemos concluir por tanto que este sistema es lo suficientemente sensible como para dar un resultado relativamente satisfactorio, pero aun así debería mejorarse en futuras aproximaciones ya que su especificidad deja mucho que desear.

4.1.3. Discusión

Para una primera iteración los resultados han sido buenos. En los árboles de decisión, kNN y SVM se han coseguido valores de sensibilidad mayores al 80 %, en algunos casos rozando el 90 %. Por otro lado, las RR.NN.AA han dado un mal resultado. Esto se debe a que las características escogidas no son las adecuadas (Figura 4).

Los mejores resultados los hemos obtenido con las SVM (mejor resultado global), kNN (segundo mejor resultado) y árboles de decisión. Los peores resultados feuron los de las RR.NN.AA ya que la sensibilidad ronda 66 % y la especificidad, el 59 %.

Estos resultados se pueden explicar con relativa facilidad. En el caso de las SVM y las kNN los datos son buenos ya que, a pesar de la gran cantidad de ruido, podemos observar 3 áreas diferenciadas en la distribución de los datos (Figura 4), de esta forma a la svm le sería sencillo separar los casos positivos a pesar del ruido. Por otro lado, tener tres zonas diferenciadas también ayuda a que knn sea capaz de determinar a que grupo pertenece un elemento. A su vez la gran cantidad de ruido explica que, por un lado, las SVM necesitan valores altos de C que acotan el area de forma más restrictiva, los kNN (salvo, curiosamente, en el caso óptimo) dan buenos resultados con un elevado número de vecinos (ya que el ruido provoca que los errores sean elevados con un número bajo), en el caso de los árboles el ruido complica la tarea y para las redes la hace directamente imposible.

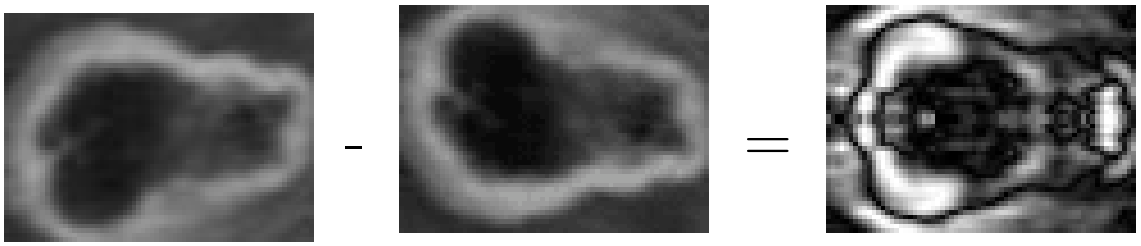
Para la siguiente iteración usaremos la simetría horizontal como técnica de extracción de características, para que así los datos de entrada no se encuentren tan mezclados y poder obtener mejores resultados a la hora de clasificar.

4.2. Segunda aproximación

4.2.1. Descripción

En esta segunda aproximación continuamos con el mismo objetivo: detectar la presencia de tumor en una imagen de TC. Las muestras usadas para el entrenamiento van a ser las mismas que en la iteración anterior.

El cambio que vamos a hacer es en la característica elegida. Ahora calcularemos la media y desviación típica del valor absoluto de la diferencia entre la imagen original y su imagen invertida en el eje X Figura 9. Con esto esperamos que aumente la diferencia entre la media y la desviacion típica de la escala de grises de las imagenes que contienen tumor y las que no.



(a) Original (b) Invertida (c) Diferencia

Figura 9: Simetría vertical con tumor

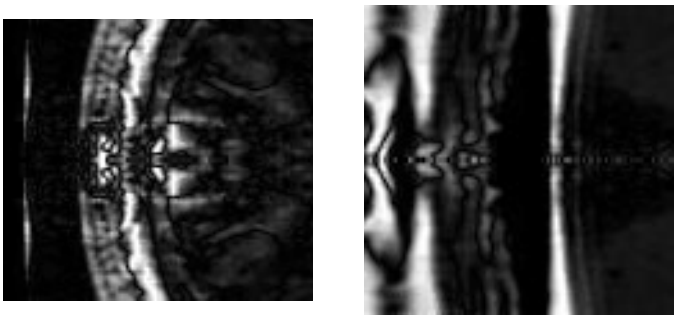


Figura 10: Ejemplos de simetrías verticales sin tumor

En este caso también utilizaremos los valores de media y desviación típica normalizados entre 0 y 1. En la Figura 11 se puede ver como están distribuidas las entradas. Como se puede ver sigue habiendo mucho ruido y no están claramente diferenciadas.

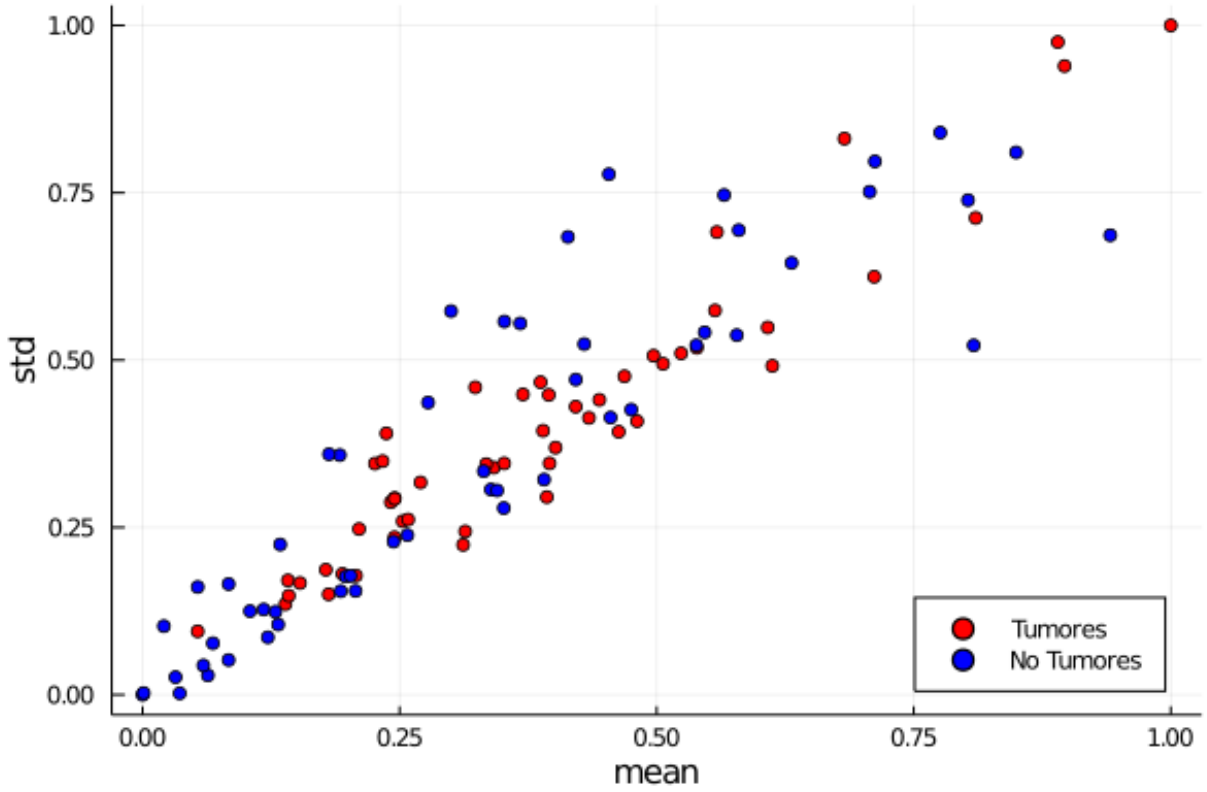


Figura 11: Distribución de los atributos del dataset

4.2.2. Resultados

- **RR.NN.AA**

  En cuanto a los parámetros de entrenamiento, no hubo modificaciones:

  1. Ratio de aprendizaje: 0.01
  2. Número máximo de ciclos de entrenamiento: 10000
  3. Número de ciclos sin mejora en la validación: 250
  4. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  5. Número de iteraciones por subconjunto: 75
  6. Ratio de elementos entrenamiento/validacion: 0.8/0.2

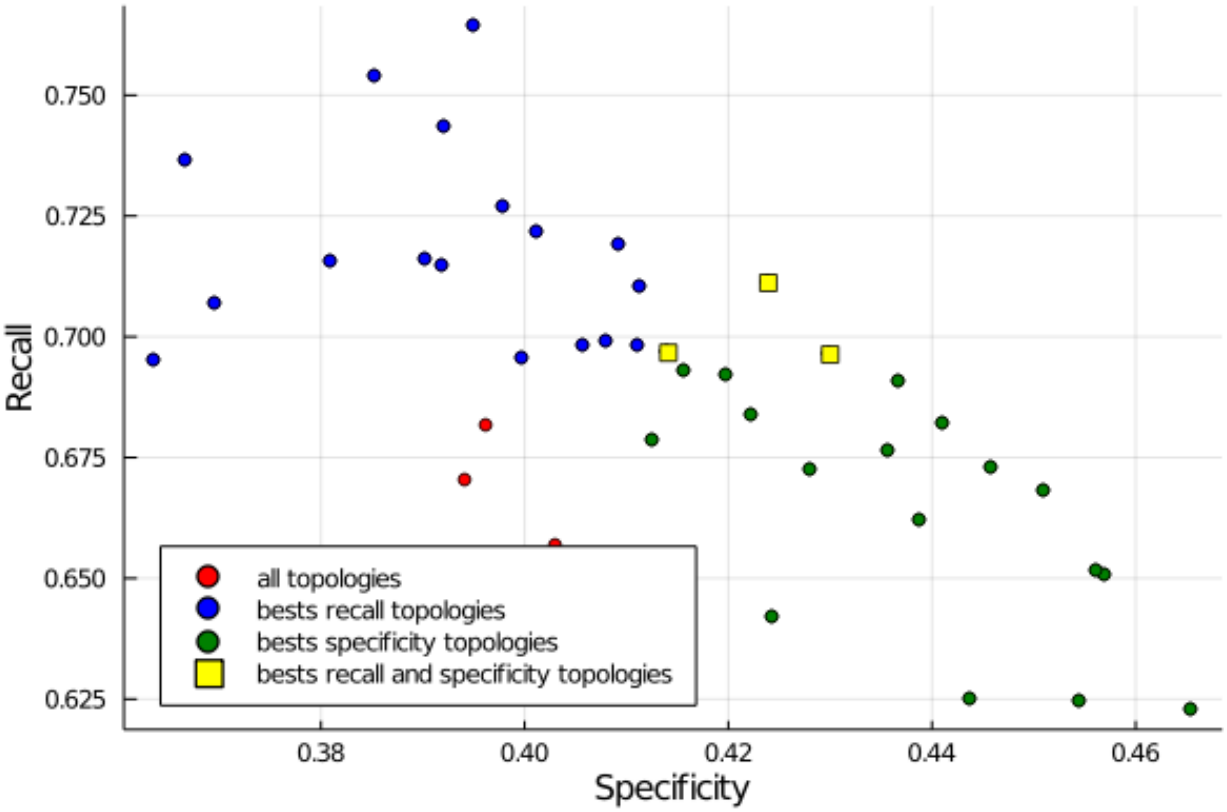


Figura 12: Valores de sensibilidad y especificidad obtenidos

Esta vez se entrenaron 40 topologías diferentes (en la anterior iteración 110). Los tres mejores resultados por sensibilidad son:

Mejores resultados		
Topología	Sensibilidad	Especificidad
[8,1]	0.7642	0.3949
[7,1]	0.7536	0.3852
[5,1]	0.7431	0.3921

Cuadro 10: Tabla de mejores resultados

Como se puede ver no hemos conseguido una mejora notable con respecto a la iteración 1. A pesar de que la sensibilidad es mayor que antes, los niveles de especificidad son demasiado bajos. Como se puede ver en la Figura 12 no se han conseguido valores alto en sensibilidad y especifidad a la vez (la especificidad nunca supera el 50 %).

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,024	1,276	5,3
	Negativo	3,338	1,962	5,3
	Total	$7,362$	3,238	N

Cuadro 11: Matriz de confusión

Como se puede observar en el Cuadro 11, la red de neuronas tiene una tendencia a clasificar todo como positivo. A pesar que en nuestro sistema es preferible la sobreestimación de positivos, este resultado es demasiado malo.

- **SVM**

Los parámetros de entrenamiento de las SMV son los mismos que en la iteración anterior:

1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
2. Cuatro tipos de kernel: lineal, polinomial, RBF y sigmoide
3. C: 0.1, 1, 10, 100, 1000, 10000
4. Degree: 3
5. Gamma: 2

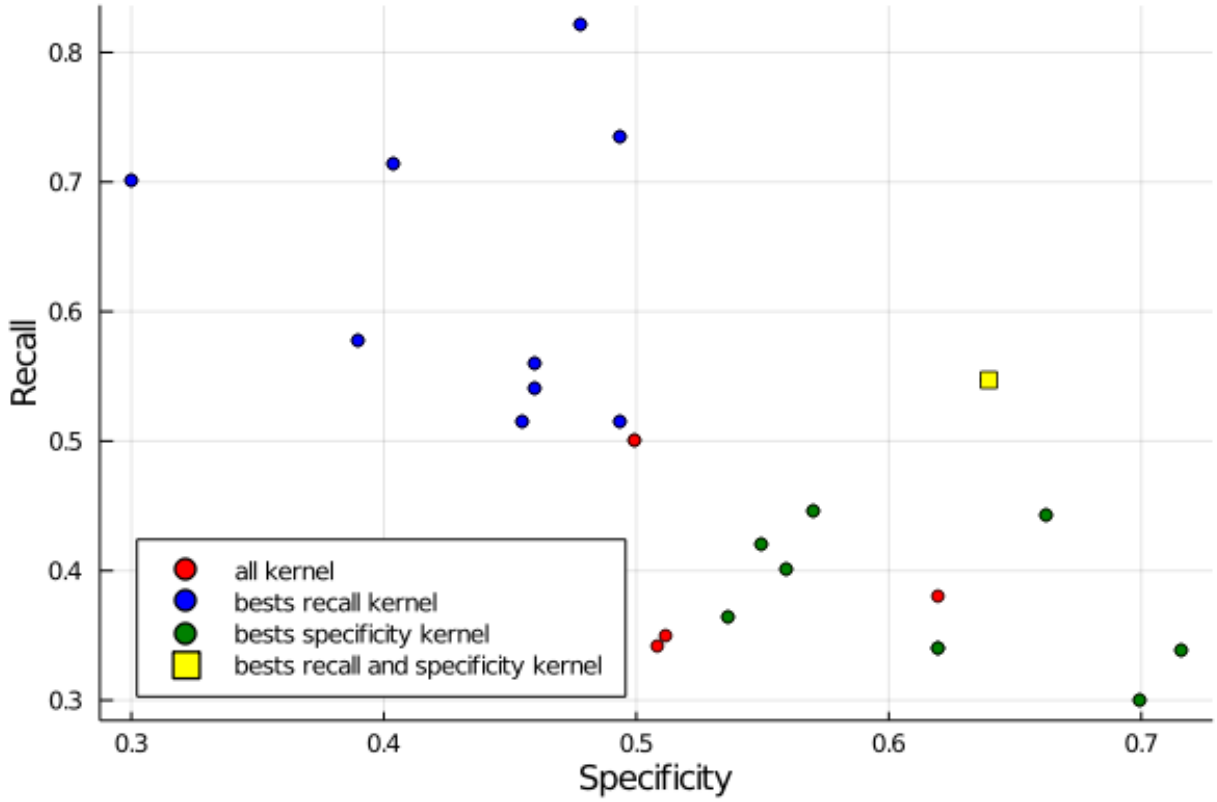


Figura 13: Valores de sensibilidad y especificidad obtenidos

La Figura 13 muestra la distribución de los top 10 mejores resultados del entrenamiento. El top 3 sería:

Mejores resultados		
Kernel y C	Sensibilidad	Especificidad
linear1000.0	0.5471	0.6396
rbf1000.0	0.7345	0.4935
rbf10000.0	0.8207	0.4779

Cuadro 12: Tabla de mejores resultados

Nuevamente los resultados han empeorado con respecto a la primera iteración. Como se puede apreciar, cuando el nivel de sensibilidad es alto, el nivel de especificidad es bajo y viceversa. Esto hace que no hay un mejor resultado claro.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	2,7	2,6	5,3
	Negativo	1,9	3,4	5,3
	Total	4,6	6	N

Cuadro 13: Matriz de confusión

En el Cuadro 13 se puede apreciar que la clasificación no es buena. Los valores falsos tienen un valor muy próximo a los valores reales. En concreto, el valor de los falsos positivos es demasiado alto.

- **Árboles de decisión**

  En cuanto a los parámetros de entrenamiento de los árboles de decisión son:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Max Depth: de 1 a 20
  3. Random State: 1

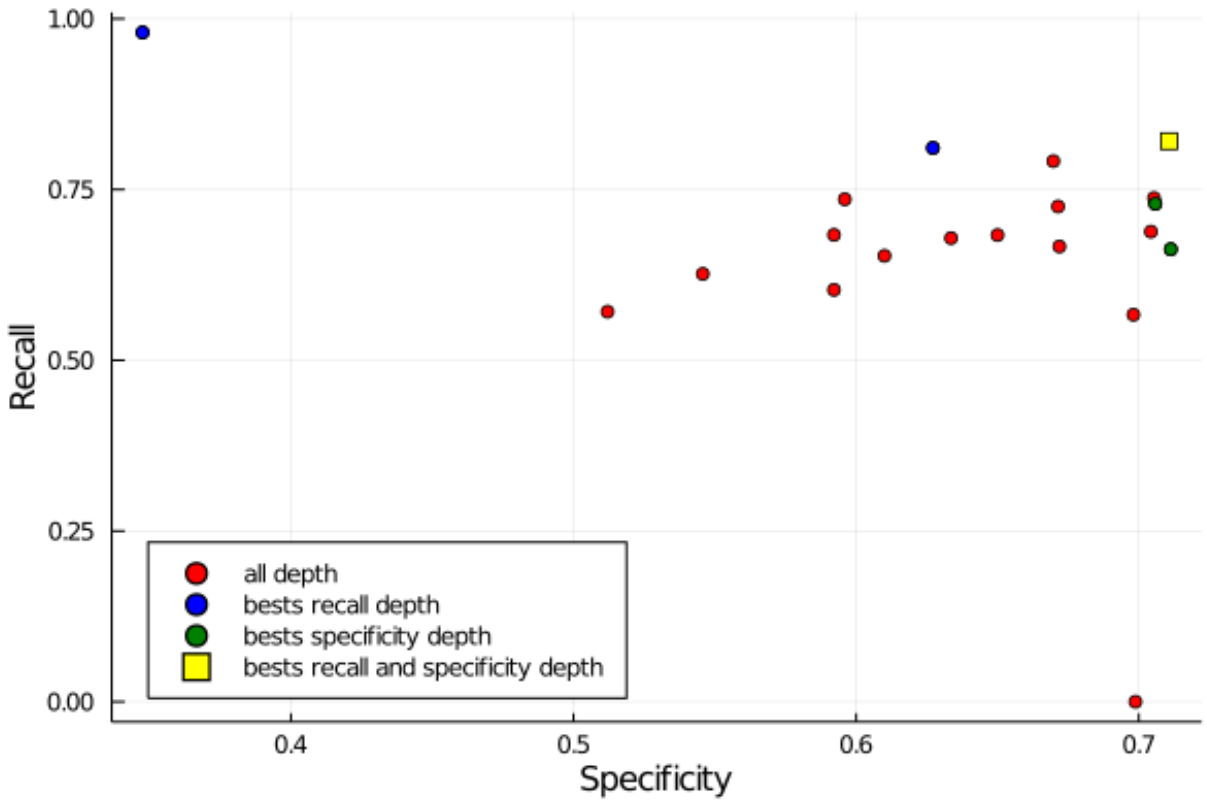


Figura 14: Valores de sensibilidad y especificidad obtenidos

En la Figura 14 se aprecia que la gran mayoría de resultados supera el 60 % de sensibilidad y especificidad. Estos resultados son semejantes a los de la iteración anterior.

Mejores resultados		
Profundidad	Sensibilidad	Especificidad
5	0.8207	0.7107
4	0.8108	0.6271
3	0.7915	0.6697

Cuadro 14: Tabla de mejores resultados

El mejor resultado entre todos ellos es el de profundidad 5, ya que tiene una sensibilidad del 82 % y una especificidad del 71 %. Le seguirían el de profundidad 4 (81 % de sensibilidad y una especificidad del 62 %) y profundidad 3 (79 % de sensibilidad y una especificidad del 66 %). Obtenemos resultados muy similares a los de la primera observación pero siguen siendo óptimos.También hacemos mención al resultado con profundidad 1 ya que supera una sensibilidad del 90 % pero baja considerablemente la especificidad hasta casi el 34 %. La matriz de confusión del mejor resultado sería la siguiente:

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,3	1,0	5,3
	Negativo	1,6	3,7	5,3
	Total	5,9	4,7	N

Cuadro 15: Matriz de confusión

- **kNN**

  Los parámetros utilizados para el algoritmo kNN son:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Número de neighbors: de 1 a 20

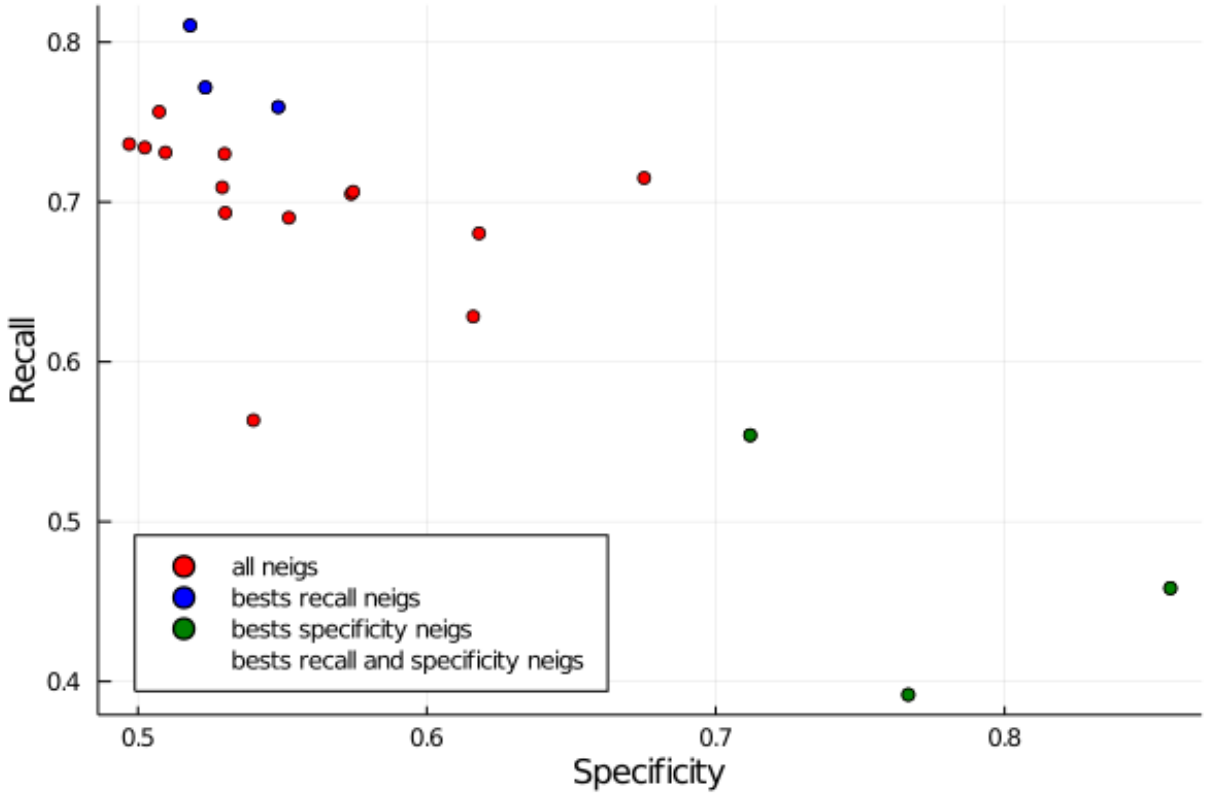


Figura 15: Valores de sensibilidad y especificidad obtenidos

En la Figura 15 se ve que la sensibilidad es alta en la mayoría de casos, mientras que la especificidad está en torno al 50 % y 60 %.

Mejores resultados		
Vecinos	Sensibilidad	Especificidad
11	0.8104	0.5180
7	0.7716	0.5233
17	0.7594	0.5485

Cuadro 16: Tabla de mejores resultados

Los mejores resultados han sido con 11 vecinos dando una sensibilidad superior al 80 % y especificiad de 51,8 %.Posteriormente le siguen los resultados con 7 y 17 vecinos con resultados de más del 75 % en sensibilidad y 52,33 % y 54,85 % de especificidad respectivamente.

Como se puede observar en el Cuadro 17, los verdaderos positivos y negativos son elevados mientras que los falsos negativos son bajos. Sin embargo, los falsos positivos siguen siendo altos.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,3	1,0	5,3
	Negativo	2,6	2,7	5,3
	Total	6,9	3,7	N

Cuadro 17: Matriz de confusión

4.2.3. Discusión

Los resultados en esta segunda iteración han sido malos ya que no se ha conseguido una mejora con respecto a la primera iteración.

Con las RR.NN.AA se ha conseguido mejor sensibilidad pero con una especificidad muy baja, esto se debe a que la RR.NN.AA está asignando positivos a la mayor parte de los patrones de forma indiscriminada. En cuanto a las SVM tanto los niveles de especificidad como los de sensibilidad son peores que en la primera iteración debido a la gran cantidad de ruido que hay. Por otro lado, los árboles de decisión han vuelto a dar los mejores resultados (sensibilidad en torno al 80 % y especificidad 70 %). Por último con kNN también hemos conseguido buenos resultados, pero en comparación con los de la primera iteración, también son malos.

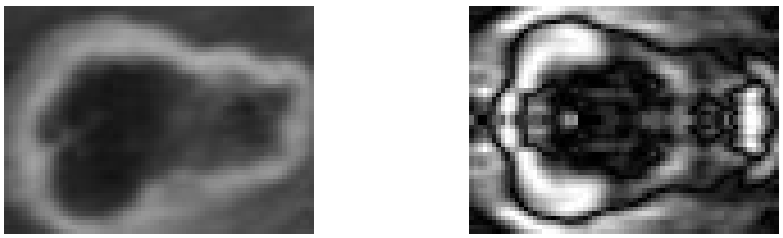
Debido a que no se ha producido ninguna mejora significativa en ninguno de los algoritmos, para la siguiente iteración usaremos 4 atributos de entradas: media y desviacion típica de la escala de grises de las imagenes originales y media y desviación típica de la escala de grises de las simetrías.

4.3. Tercera aproximación

4.3.1. Descripción

En esta tercera aproximación seguiremos con el objetivo de detectar sola y exclusivamente la presencia de tumor en una imagen de TC. Las muestras usadas para el entrenamiento van a ser las mismas que en la iteración anterior.

El cambio que vamos a hacer es en las características elegidas. Ahora calcularemos la media y desviación típica como en la aproximación 1 y estos mismos atributos del valor absoluto de la diferencia entre la ventana y su imagen invertida en el eje X (Figura 16). Se ha elegido esta mezcla de mediciones para intentar encontrar mejores resultados.



(a) Original (b) Imagen simétrica

Figura 16: Imagenes recortadas

Como ahora hay 4 entradas, hay 4 dimensiones, y por lo tanto se usará una tabla para su representación (Cuadro 18). Se puede ver un ejemplo de como quedan distribuidos los datos de entrada.

Tabla inputs				
Column	Mean	Std	MeanInv	StdInv
1	0.1992	0.2814	0.3958	0.3456
2	0.1597	0.2955	0.4341	0.4136
...	...	...	...	...
105	0.2420	0.2703	0.3000	0.5731
106	0.1023	0.4254	0.6311	0.6451

Cuadro 18: Tabla entradas del sistema

4.3.2. Resultados

- **RR.NN.AA**

  Volvimos a reducir el número de topologías a entrenar a 20. Los parametros de entrenamiento no cambian:

  1. Ratio de aprendizaje: 0.01
  2. Número máximo de ciclos de entrenamiento: 10000
  3. Número de ciclos sin mejora en la validación: 250
  4. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  5. Número de iteraciones por subconjunto: 75
  6. Ratio de elementos entrenamiento/validacion: 0.8/0.2

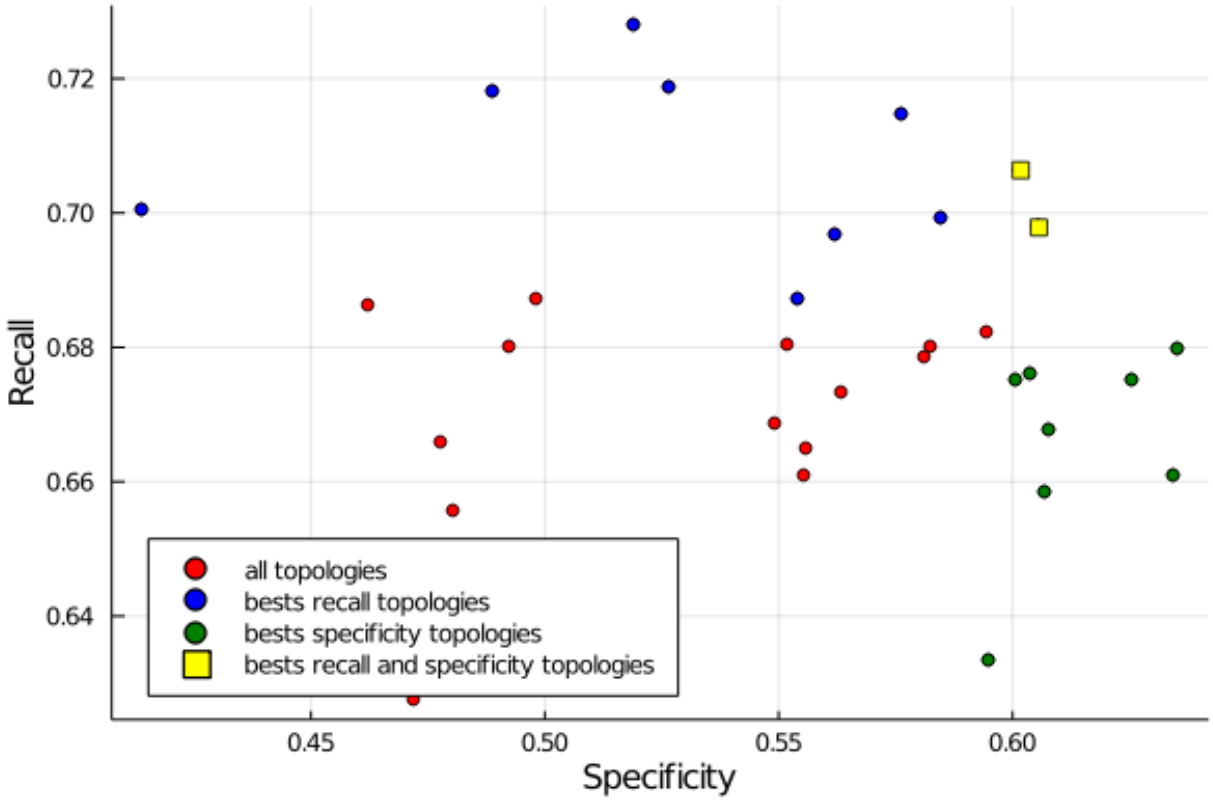


Figura 17: Valores de sensibilidad y especificidad obtenidos

Como se puede apreciar en la Figura 17 los resultados han mejorado bastante. Hay varios resultados con más de 70 % de sensibilidad y 50 % de especificidad. Los mejores tres resultados son:

Mejores resultados		
Topología	Sensibilidad	Especificidad
[6,5]	0.7063	0.6017
[7,5]	0.7147	0.5762
[5,5]	0.6608	0.6342

Cuadro 19: Tabla de mejores resultados

Por primera vez se supera el 66 % en sensibilidad de la primera aproximación, además de mantener el nivel de especificación. Para la topología [6,5] se ha sacado la siguiente matriz de confusión (Cuadro 20):

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	3,58	1,71	5,29
	Negativo	2,17	3,13	5,3
	Total	5,75	4,84	N

Cuadro 20: Matriz de confusión

En la matriz podemos ver como los valores verdaderos son superiores a los valores falsos. Además los valores de positivos falsos son mayores que los falsos negativos, lo cuál es bueno. Con esto se deduce que esta nueva característica beneficia a las RR.NN.AA.

- **SVM**

  Como parámetros de entrenamiento de las SMV utilizamos:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Cuatro tipos de kernel: lineal, polinomial, RBF y sigmoide
  3. C: 0.1, 1, 10, 100, 1000, 10000
  4. Degree: 3
  5. Gamma: 2

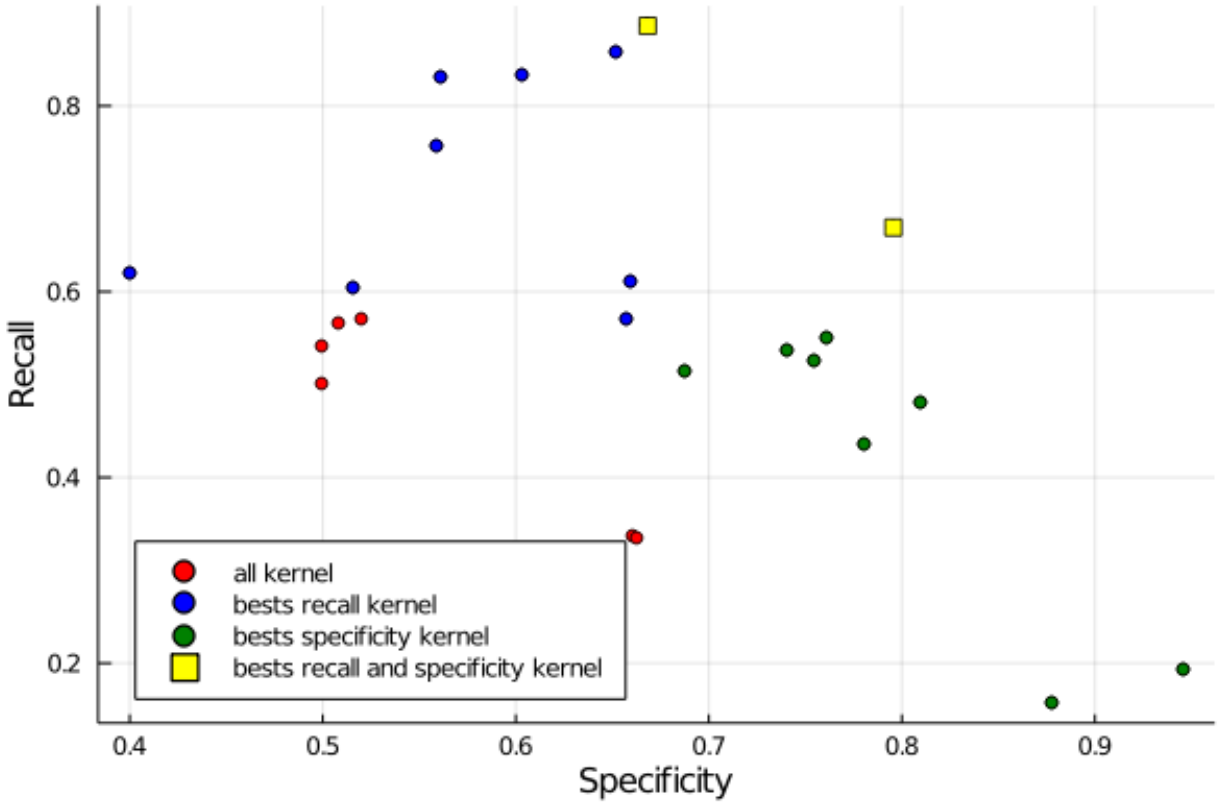


Figura 18: Valores de sensibilidad y especificidad obtenidos

En este entrenamiento conseguimos buenos resultados, ya que conseguimos valores del 80 % de sensibilidad y una especificidad de en torno al 70 % (Figura 18). Los mejores resultados son los siguientes:

Mejores resultados		
Kernel y C	Sensibilidad	Especificidad
poly10000.0	0.8861	0.6683
rbf1000.0	0.8571	0.6523
rbf10000.0	0.8316	0.6036
poly1000.0	0.6690	0.7955

Cuadro 21: Tabla de mejores resultados

Claramente lo que mejor resultado da es el kernel polinomial y RBF con valores muy altos de C. El Cuadro 22 muestra la matriz de confusión del mejor resultado del

entrenamiento (poly10000.0). Tanto el nivel de verdaderos positivos como de verdaderos negativos es bastante más grandes que los valores falsos. Nuevamente los falsos negativos son muy bajos. Esto reafirma la efectividad de la nueva característica.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,6	0,7	5,3
	Negativo	1,8	3,5	5,3
	Total	6,4	4,2	N

Cuadro 22: Matriz de confusión

- **Árboles de decisión**

  En cuanto a los parámetros de entrenamiento de los árboles de decisión son:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Max Depth: de 1 a 20
  3. Random State: 1

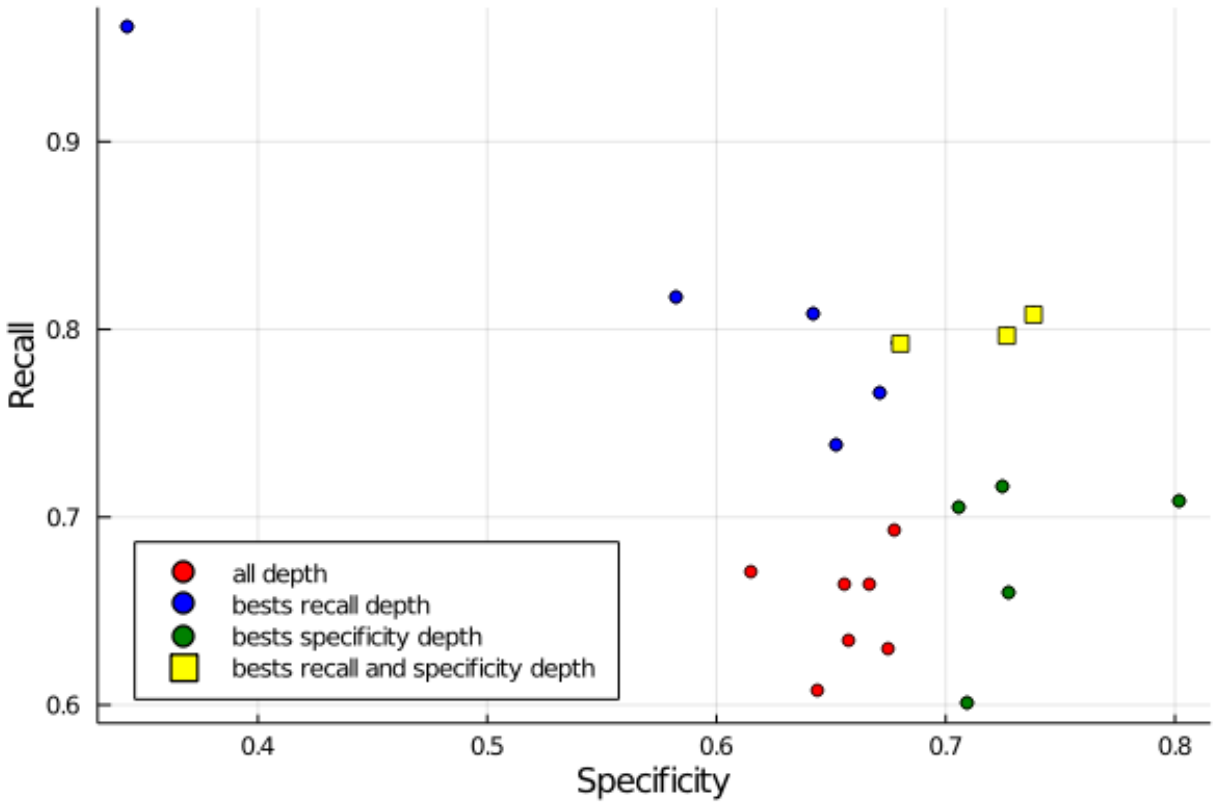


Figura 19: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Profundidad	Sensibilidad	Especificidad
4	0.8078	0.7383
7	0.7966	0.7266
6	0.7923	0.6802

Cuadro 23: Tabla de mejores resultados

De nuevo se obtienen resultados muy buenos (Figura 19). Sensibilidad del 80 % y especificidad del 70 %. Con respecto al resto de iteraciones no hay mucha diferencia, sólo sube ligeramente la especificidad.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,2	1,1	5,3
	Negativo	1,5	3,8	5,3
	Total	5,7	4,9	N

Cuadro 24: Matriz de confusión

En el Cuadro 24 (matriz de confusión del árbol con depth 4) podemos apreciar la buena clasificación del sistema. Volvemos a conseguir valores reales altos y falsos bajos, asi como unos falsos positivos más altos que los negativos.

- **kNN**

  Los parámetros utilizados para el algoritmo kNN son:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Número de neighbors: de 1 a 20

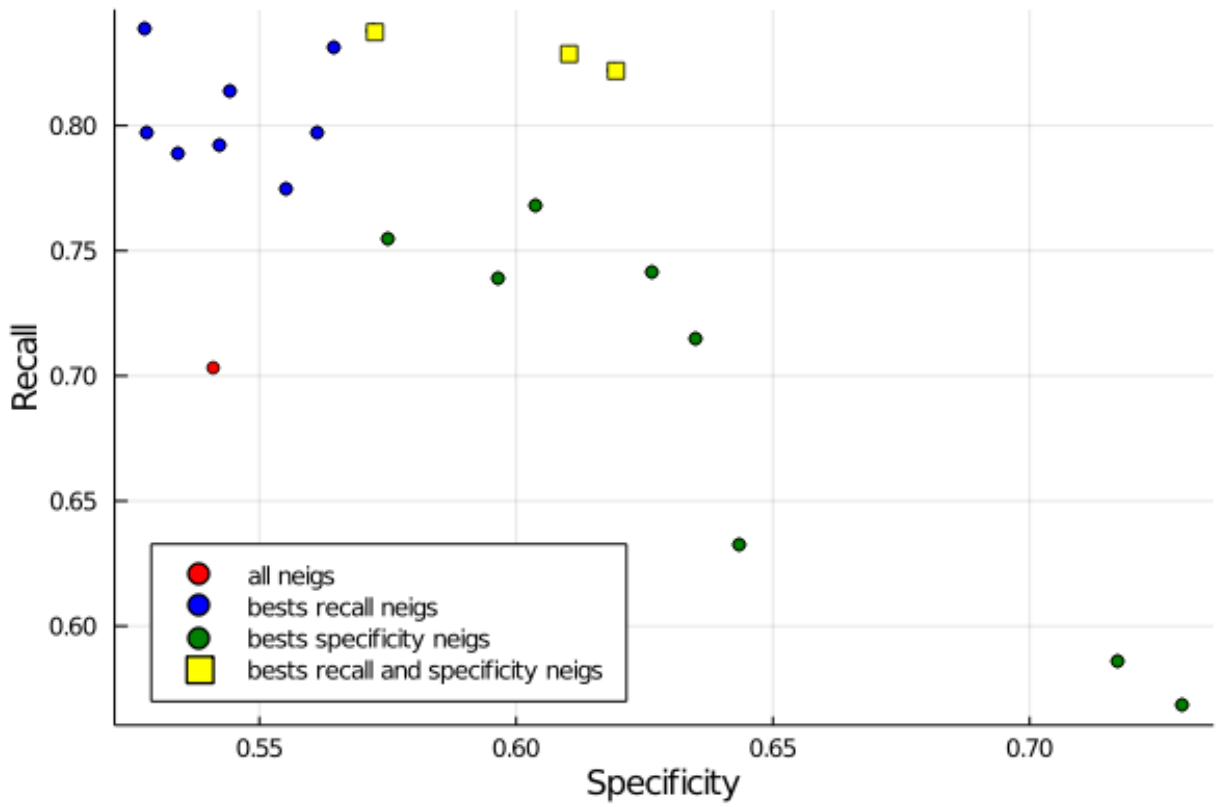


Figura 20: Valores de sensibilidad y especificidad obtenidos

Otra vez vuelven a haber valores de 80 % en sensibilidad y 60 % de especificidad. La gran mayoría de resultados tienen sensibilidad alta pero aun no hay ninguno que tenga un porcentaje tanto sensibilidad como de especificidad altos. Los mejores tres resultados son:

Mejores resultados		
Vecinos	Sensibilidad	Especificidad
10	0.8286	0.6103
3	0.8219	0.6194
17	0.8373	0.5725

Cuadro 25: Tabla de mejores resultados

Para el kNN de 10 vecinos, se muertra en la Figura 26 su matriz de confusión. Al igual que con el resto de algoritmos en esta aproximación los valores reales son altos. Sin embargo la cantidad de falsos positivos es un poco elevada.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,2	1,1	5,3
	Negativo	2,1	3,2	5,3
	Total	6,3	4,3	N

Cuadro 26: Matriz de confusión

4.3.3. Discusión

Los resultados de esta aproximación son mucho mejores que los de la segunda aproximación, pero similares que los de la primera. Esta mejora es debido al cambio de las características, ya que ahora tenemos menos ruido que antes al trabajar con un número mayor de atributos.

Los mejores resultados se han conseguido con las SVM, árboles de decisión y kNN. Las SVM han alcanzado la sensibilidad más alta (88 %) manteniendo un nivel bueno de especificidad (66 %). Los árboles y kNN dan resultados muy similares; los kNN tienen una sensibilidad ligeramente mayor (80 % frente a 82 %), mientras que los árboles los superan en la especificidad (61 % frente a 70 %). Por último volvemos a tener a las RR.NN.AA las cuales apenas llegan al 70 % de sensibilidad, con una especificidad de 60 %.

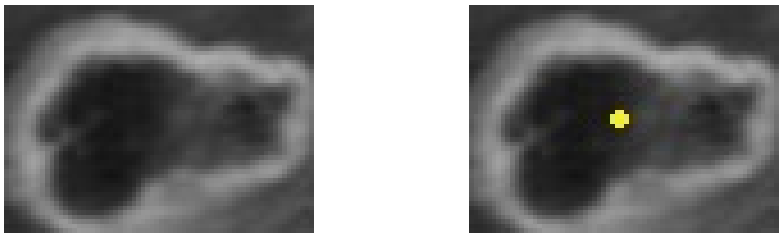
Para la cuarta aproximación nos centraremos en el uso de centroides en las imágenes.

4.4. Cuarta aproximación

4.4.1. Descripción

Debido a los resultados obtenidos anteriormente, para esta cuarta aproximación nos mantendremos con el mismo objetivo de detectar tumores en una imagen. Las muestras usadas en el entrenamiento tampoco varían.

En cuanto a las características extraídas añadiremos los centroides. Estos son puntos que se colocan en el centro de una figura. Por ejemplo, el centroide de un triángulo sería el punto en el que coinciden las tres medianas. Con ellos calcularemos la distancia que hay desde este elemento con los bordes de la imagen (Figura 21). De esta forma, el sistema podrá detectar las formas redondas y ovaladas que tienen los tumores.



(a) Original (b) Centroide

Figura 21: Imagenes recortadas

Los datos de entrada con estas características serían los del Cuadro 27. Volvemos a utiizar la media y desviación típica como en la aproximación 1 y 3, y a mayores usamos las coordenadas del centroide en la imagen.

Tabla inputs				
Column	Mean	Std	coordenada X	coordenada Y
1	0.1992	0.2814	0.0915	0.1351
2	0.1597	0.2955	0.1901	0.1081
...	...	...	...	...
105	0.2420	0.2703	0.0211	0.2522
106	0.1023	0.4254	0.2112	0.4144

Cuadro 27: Tabla entradas del sistema

4.4.2. Resultados

- **RR.NN.AA**

  Los parametros de entrenamiento y el número de topoogías no cambian no cambian con respecto a la iteración anterior:

  1. Ratio de aprendizaje: 0.01
  2. Número máximo de ciclos de entrenamiento: 10000
  3. Número de ciclos sin mejora en la validación: 250
  4. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  5. Número de iteraciones por subconjunto: 75
  6. Ratio de elementos entrenamiento/validacion: 0.8/0.2

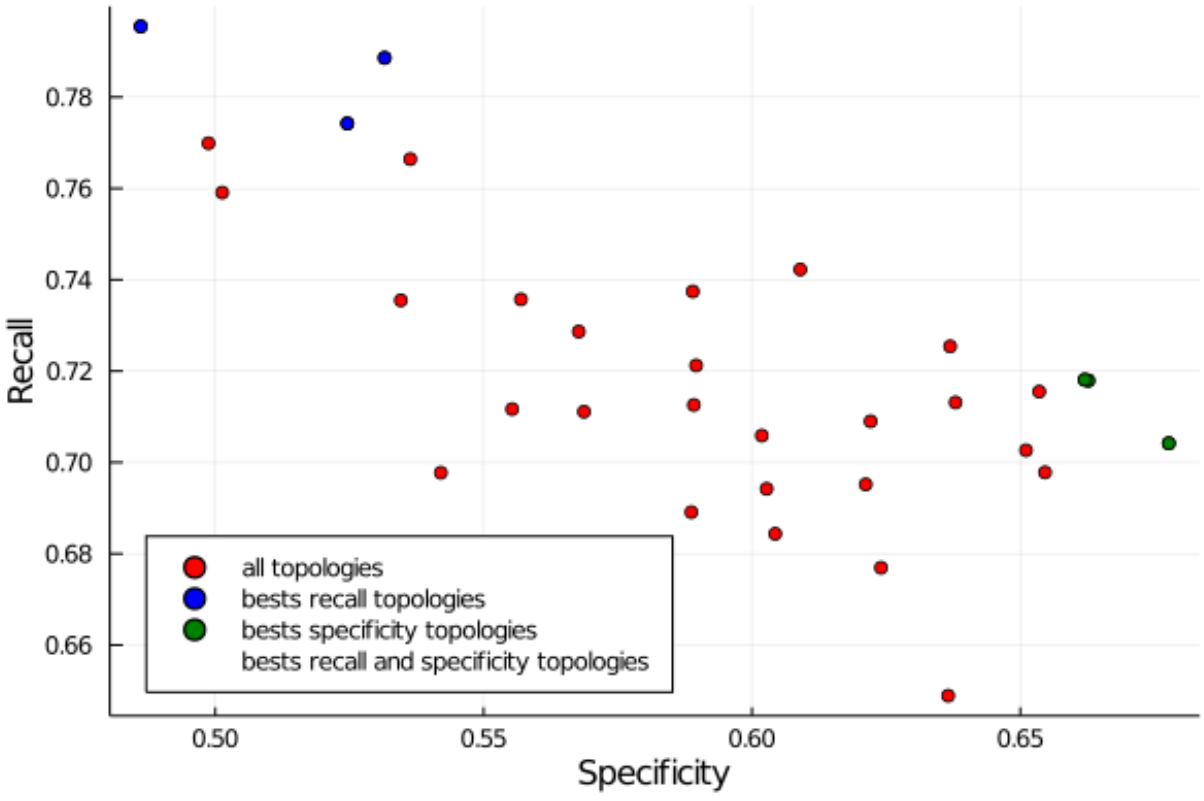


Figura 22: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Topología	Sensibilidad	Especificidad
[4,1]	0.7883	0.5315
[5,1]	0.7664	0.5363
[6,3]	0.7422	0.609

Cuadro 28: Tabla de mejores resultados

En el Cuadro 28 podemos ver que los resultados mejoran ligeramente respecto a la primera y tercera iteración, aunque no lo suficiente. Como mejores resultados conseguimos sensibilidades del 74-79 % y especificidades del 53-61 %. En los casos en los que se consigue una sensibilidad más alta, la especificidad cae, pero vemos que salvo en un par de casos logramos que esta se mantenga siempre sobre el 50 %. No obstante, no podemos considerar que estas sean unas buenas métricas ya que si queremos obtener especificidades superiores al 65 % debemos dejar caer la sensibilidad hasta el 72 %

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	3,70	1,59	5,29
	Negativo	1,97	3,33	5,3
	Total	5,67	4,92	N

Cuadro 29: Matriz de confusión

El Cuadro 29 muestra la matriz de confusión de la topología [4]. Se obtuvo mejores resultados que en la anterior iteración ya que el número de falsos positivos es menor en comparación. Aún asi, las RR.NN.AA siguen sin dar buenos resultados.

- **SVM**

  Como parámetros de entrenamiento de las SMV utilizamos:

  1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)
  2. Cuatro tipos de kernel: lineal, polinomial, RBF y sigmoide
  3. C: 0.1, 1, 10, 100, 1000
  4. Degree: 45
  5. Gamma: 2

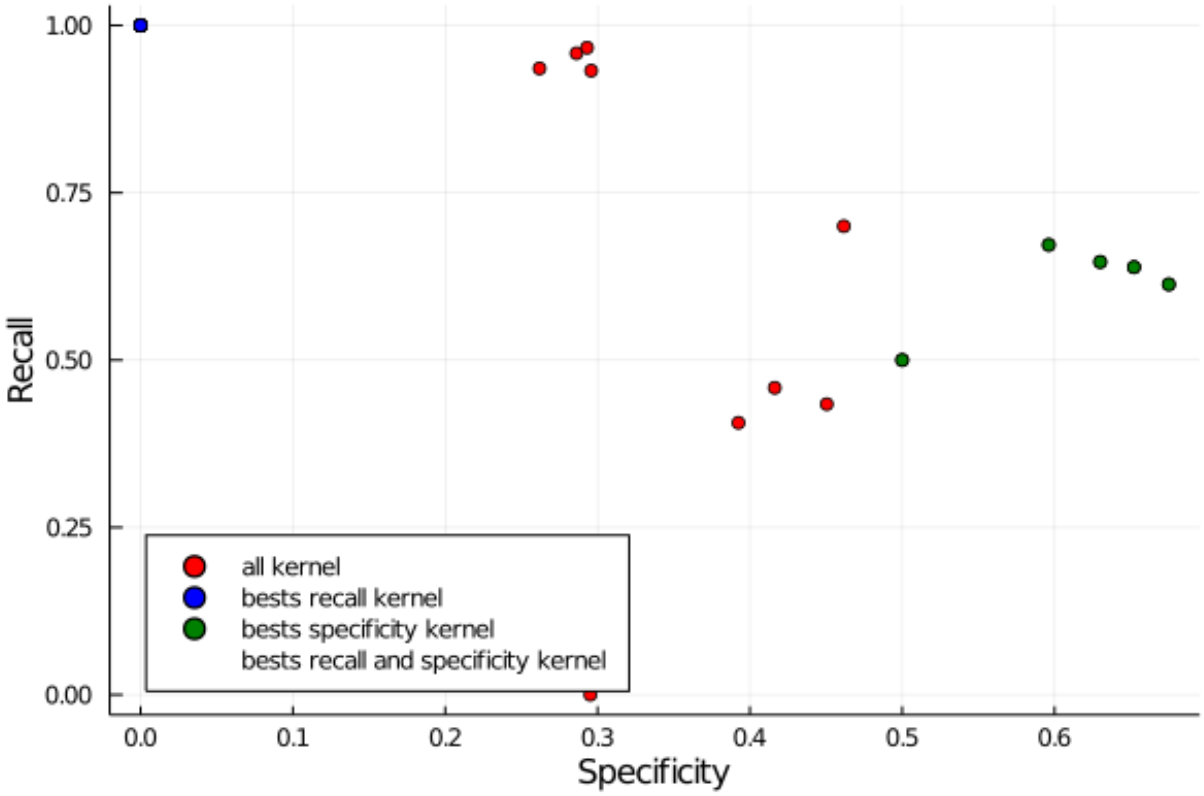


Figura 23: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Kernel y C	Sensibilidad	Especificidad
linear0.1	0.64	0.63
linear1.0	0.67	0.59
linear100.0	0.63	0.65

Cuadro 30: Tabla de mejores resultados

Como se puede ver en el Cuadro 30, el mejor kernel ha sido el lineal. También se puede apreciar que los resultados han sido peores que en la aproximación anterior. Las sensibilidades y especificidades rondan el 60 %, estando parejas en la mayoría de los casos. Hay casos extremos en los que la sensibilidad era de más del 90 %, pero la especificidad era inferior al 30 %. Cabe destacar el alto valor que tuvimos que utilizar para el ángulo del kernel polinomial, ya que con los valores utilizados en anteriores

aproximaciones la estimación de tiempo para el entrenamiento en nuestros equipos era de 16 horas (solo para kernel polinomial con C=100).

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	3,4	1,9	5,3
	Negativo	2,0	3,3	5,3
	Total	5,4	5,2	N

Cuadro 31: Matriz de confusión

El Cuadro 31 es la matriz de confusión del resultado con kernel lineal y C = 0.1. La proporcion de valores positivos es mayor que la de negativos, pero no en una gran medida. Por lo tanto estos resultados no son buenos.

- **Árboles de decisión**

En cuanto a los parámetros de entrenamiento de los árboles de decisión son:

1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)

2. Max Depth: de 1 a 20

3. Random State: 1

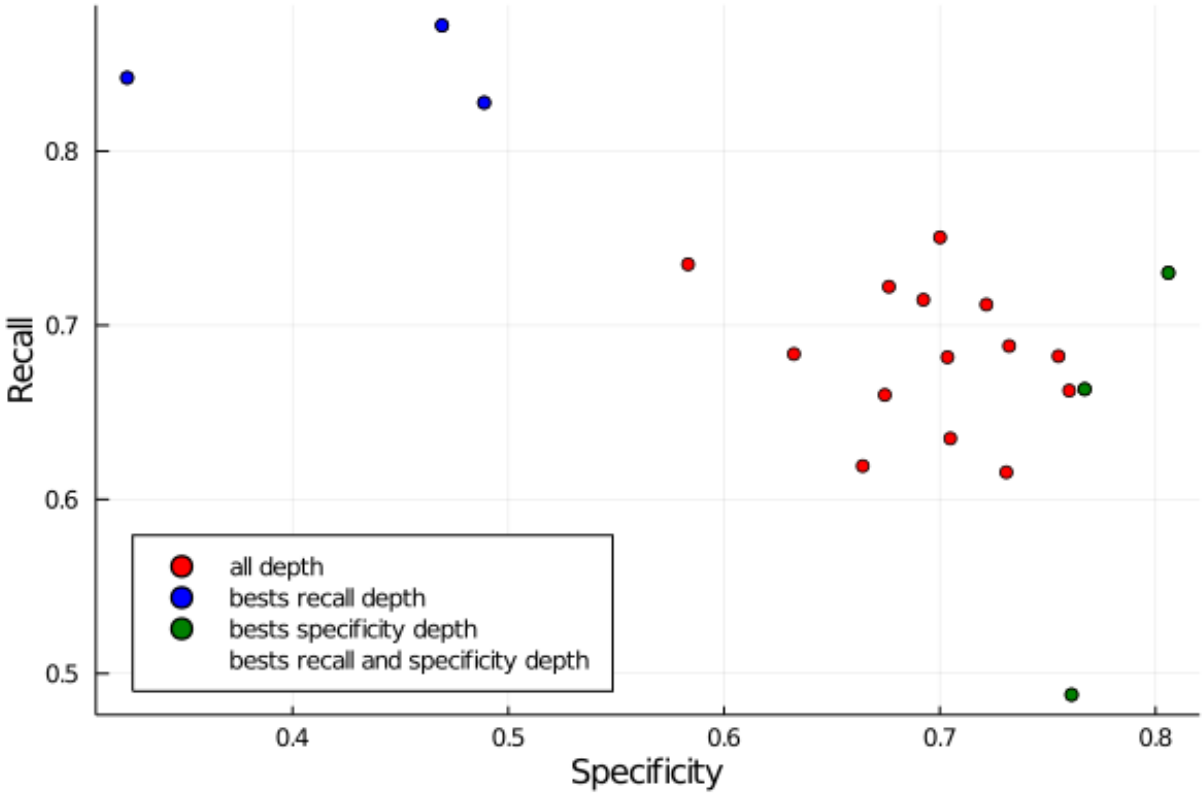


Figura 24: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Profundidad	Sensibilidad	Especificidad
19	0.7504	0.7001
14	0.7301	0.8059
15	0.7221	0.6764

Cuadro 32: Tabla de mejores resultados

Como se puede apreciar en la figura **??**) obtenemos sensibilidad del 75 % y especificidad del 70 %. Por tanto obtenemos peores resultados que en la aproximación anterior.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	3,9	3,5	7,4
	Negativo	1,8	1,4	3,2
	Total	5,7	4,9	N

Cuadro 33: Matriz de confusión

En el Cuadro 33 (matriz de confusión del árbol con depth 19) podemos apreciar buenso resultados. Conseguimos valores reales altos y falsos bajos, asi como unos falsos positivos más altos que los negativos.

- **kNN**

Los parámetros utilizados para el algoritmo kNN son:

1. Número de subconjuntos de cross-validation: 10 (12 elementos por subconjunto)

2. Número de neighbors: de 1 a 20

Encontramos valores de 87 % en sensibilidad y 58 % de especificidad. La gran mayoría de resultados tienen sensibilidad mas alta que en la anterior aproximación.Los mejores tres resultados son:

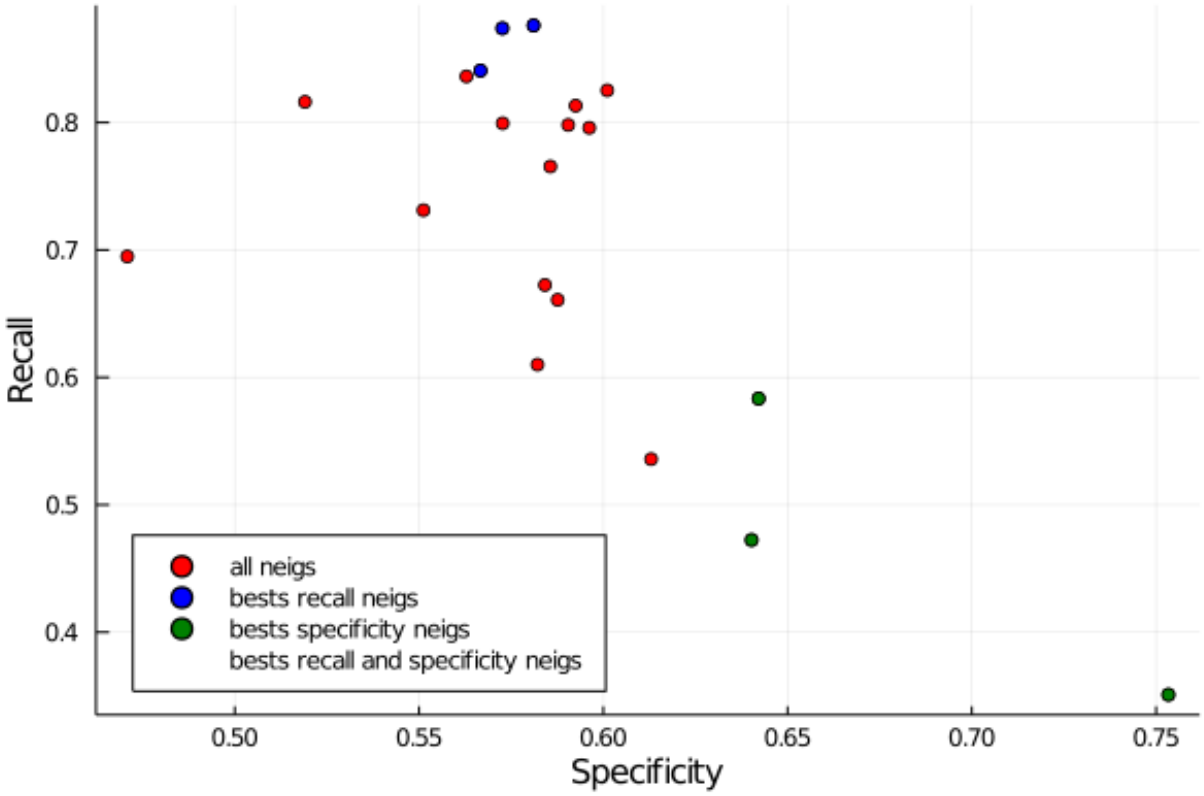


Figura 25: Valores de sensibilidad y especificidad obtenidos

Mejores resultados		
Vecinos	Sensibilidad	Especificidad
13	0.8761	0.5810
15	0.8739	0.5726
17	0.8407	0.5666

Cuadro 34: Tabla de mejores resultados

Para el kNN de 13 vecinos, se muestra en la Cuadro 35 su matriz de confusión.Se comprueba que tenemos poca cantidad de negativos con respecto a otras aproximaciones y alto de postivos.

		Valores predecidos		
		Positivo	Negativo	Total
Valores reales	Positivo	4,5	2,9	7,4
	Negativo	2,4	0,8	3,2
	Total	6,9	3,7	N

Cuadro 35: Matriz de confusión

4.4.3. Discusión

Con RR.NN.AA seguimos sin superar el 70 % de sensibilidad con un nivel alto de especificidad, y aunque sí conseguimos una pequeña mejoría, esta sigue siendo totalmente insuficiente. En cuanto a las SVM hemos conseguido peores resultados que en la iteración 1 y 3. Los árboles de decisión son el algoritmo con el que hemos conseguido mejores resultados (80 % sensibilidad y 70 % especificidad). Por último con KNN hemos conseguido niveles buenos de sensibilidad (80 %) pero con peores valores en la especificidad con respecto a la anterior iteración.

Teniendo todo esto en cuenta, los resultados no han sido buenos. Por lo tanto para la siguiente iteración dejaremos de usar estos algoritmos y pasaremos a trabajar con deep learning.

4.5. Aproximación Deep Learning

4.5.1. Descripción

Finalmente realizaremos una aproximación con deep learning. El deep learning es un tipo de machine learning con mucho auge en la actualidad. Se puede utilizar para clasificar imágenes, reconocer el habla, detectar objetos o describir contenido. Sistemas muy populares, como pueden ser Siri o Cortana, hacen uso del deep learning.

A la hora de trabajar con los algoritmos de las anteriores aproximaciones, era necesario hacer recortes de los tumores (Figura 21) para que el sistema los detectara como positivos. Esto ya no va a ser necesario ya que usaremos Redes de Neuronas Convolucionales.

Las Redes de Neuronas Convolucionales son un tipo de red donde sus neuronas corresponden a campos receptivos de una manera muy similar a las neuronas en la corteza visual primaria (V1) en un cerebro biológico. Es necesario pasarle como entrada la matriz de pixeles de la imagen, por lo tanto, hay que hacer un resize de las fotos que utilizamos ya que todas las imágenes tienen que tener el mismo tamaño. El tamaño de las imágenes va a ser de 100x100.

Además, al contrario que en las aproximaciones anteriores, no usaremos la validación cruzada debido a la alta complejidad de estas redes. Se ha realizado el entrenamiento con 8 topologías diferentes.

4.5.2. Resultados

Topologia	Conv. 1	MaxPool 1	Conv. 2	MaxPool 2	Conv. 3	MaxPool 3
1	(3,3), 1 ⇒ 16	(2,2)	(3,3), 16 ⇒ 32	(2,2)	(3,3), 32 ⇒ 32	(2,2)
2	(3,3), 1 ⇒ 20	(2,2)	(3,3), 20 ⇒ 40	(2,2)	(3,3), 40 ⇒ 40	(2,2)
3	(3,3), 1 ⇒ 8	(2,2)	(3,3), 8 ⇒ 16	(2,2)	(3,3), 16 ⇒ 16	(2,2)
4	(4,4), 1 ⇒ 16	(2,2)	(4,4), 16 ⇒ 32	(2,2)	(4,4), 32 ⇒ 32	(2,2)
5	(2,2), 1 ⇒ 32	(2,2)	(2,2), 32 ⇒ 64	(2,2)	(2,2), 64 ⇒ 64	(2,2)
6	(2,2), 1 ⇒ 16	(2,2)	(2,2), 16 ⇒ 32	(2,2)	(2,2), 32 ⇒ 32	(2,2)
7	(4,4), 1 ⇒ 20	(2,2)	(4,4), 20 ⇒ 40	(2,2)	(4,4), 40 ⇒ 40	(2,2)
8	(4,4), 1 ⇒ 16	(2,2)	(4,4), 16 ⇒ 32	(2,2)	(4,4), 32 ⇒ 32	(2,2)

Cuadro 36: Topologías utilizas en deep learning

En el Cuadro 36 se puede ver la configuración de las diferentes topologías. Como solo tenemos dos clases, la última capa de la RNA va a ser siempre del tipo Dense(x, 2, sigma). Además la tasa de aprendizaje es de 0.01.

Las columnas de Conv son las capas de convolución. El parámetro (X,Y) indica el tamaño del filtro de convolución, mientras que el Z =¿K indica los canales de entrada (como es escala de grises el primero siempre es Z = 1) y los canales de salida (K). Como función de transefencia utilizamos relu". Por otro lado, las columnas de MaxPool indican la función MaxPool, la cual consta de un par (A,B) donde A sería el número por el que se divide la matriz en el eje X, mientras que B sería lo mismo pero para el eje Y. Por ejemplo un MaxPool(2,2) pasa una imagen de 28x28 a 14x14.

Topologia	Precision
1	62.35 %
2	52.38 %
3	47.62 %
4	42.85 %
5	47.62 %
6	71.43 %
7	42.86 %
8	47.62 %

Cuadro 37: Resultados deep learning

El Cuadro 37 muestra los resultados del entrenamiento para las diferentes topologías antes definidas. Claramente los resultados no han sido buenos. El mayor porcentaje de precisión se ha conseguido con las topologías y 6, consiguiendo un 62 % y 71 % respectivamente. Además, el valor de la precisión de entrenamiento era muy bajo, en torno al 50 %, y siempre que este subía hacía bajar al porcentaje de test.

4.5.3. Discusión

Como se ha podido ver en el Cuadre 37, los resultados han sido mucho más malos de lo que nos esperábamos. Quitando la topología 6 que ha llegado al 71 % de hacierto (que ni siquiera es mucho), la gran mayoría de topologías no llegan al 50 %. Esto indica que la clasificación no se está haciendo adecuadamente ya que son números similares a los conseguidos lazando una moneda al aire.

Dejando esto de lado los canales de entrada que mejor han funcionado son los 1 =¿16, 16 =¿32 y 32 =¿32, ya que los mejores resultados (topología 1 y 6) se han conseguido con ellos.

Claramente el problema se hubica en la imágenes que hemos usado para realizar el deep learning. Quizás esto se podría corregir con aumentando el tamaño de la cantidad de imágenes. Además de esto, también se podrían probar otros tamaños para el resize en vez de 100x100.

5. Conclusiones

Las principales dificultades fueron enfrentarse al lenguaje de programación Julia desde cero (así como implementar codigo ejecutable en linux y windows) y dar con atributos de entradas para las RR.NN.AA que mejorarán algo los resultados, ya que en la mayoría de casos no conseguíamos avances significativos.

En la primera aproximación conseguimos, junto con la tercera, los mejores resultados. La características elegidas fueron la media y desviación típica de la imagen. En promedio llegamos a una sensibilidad del 80 % con una especificidad del 65 %.

El la segunda aproximación decidimos calcular la media y desviación típica del valor absoluto de la diferencia entre la imagen original y su imagen invertida en el eje X. En cuanto a los resultados, fueron muy malos. En la mayoría de los casos se mantenía la sensibilidad obtenida en la primera iteración, pero con menor especificidad, o bajaba mucho tanto la tasa de sensibilidad.

Viendo estos resultados, en la tercera aproximación hicimos una mezcla de diferentes características. Tomamos la media y desviación típica como en la aproximación 1 y estos mismos atributos del valor absoluto de la diferencia entre la ventana y su imagen invertida en el eje X. Gracias a este cambio, volvimos a obtener unos resultados muy parecidos a los de la primera aproximación.

Finalmente para las características de la cuarta aproximación nos decantamos por el uso de centroides. De esta forma calcularíamos la distancia que hay desde este elemento con los bordes de la imagen. A pesar de que los resultados no mejoraron a los anteriores, tampoco fueron malos del todo.

En cuanto a los algoritmos utilizados, el peor es claramente las RR.NN.AA, ya que se obtenía un 78 % de sensibilidad en los mejores resultados. Por otro lado, tanto las SVM y KNN han dado resultados muy parecidos (Por lo general sensibilidad de más del 80 %), pero el nivel de especificidad a veces era bajo. Los árboles de decisión han sido los que mejores resultados han dado en las cuatro primeras aproximaciones, ya que siempre se ha conseguido una sensibilidad de 80 % junto a una especificidad del 70 %.

Por otro lado, en la aproximación con deep learning nos llevamos una sopresa debido a los malos resultados del entrenaiento. Aunque se consiguió una precisión del 71 % en una ocasión, el resto de resultados fueron muy malos, ya que no llegaban ni al 50 % en algunas ocasiones. Claramente, con este porcentaje de precisión el sistema no está clasificando correctamente por lo que habría que repasar nuevamente la base de datos de las imágenes con las que estamos trabajando.

En conclusión, los resultados no han sido tan buenos como pensábamos que iban a ser a simple vista. A pesar de que un ojo humano inexperto es capaz de detectar sin problema los tumores cerebrales del dataset no hemos sido capaces de dar con unos atributos que definan las imagenes de forma que el sistema sea capaz de distinguir con alta tasa de acierto la presencia de tumores. Esto hace que no hayamos podido conseguir otros objetivos como podría ser la distinción entre tres tipos de tumores (Figura 2).

6. Trabajo futuro

La detección temprana de un cáncer con lleva, en el 90 % de los casos, a una posibilidad de curarse del mismo. Por este motivo, nuestro sistema sería muy útil para poder llevar a cabo esta tarea, ya que no habría la necesidad de que un sanitario tuviese que pasarse horas mirando como cargan las tomografías en la pantalla, además de proporcionarle una segunda opinión en la interpretación de los resultados.

Para poder mejorar los porcentajes de sensibilidad y especificidad actuales, necesitaríamos de una base de datos más grande y con más variedad de imágenes. Además de esto, sería conveniente contar de más tiempo para poder realizar los ajustes apropiados y así evitar desajustes como en la iteración 2 o en la 5, con el deep learning.

Este trabajo se podría ampliar utilizando distintos atributos (centrándose en otros aspectos de la morfología de la imagen) y aumentando la dimensión del problema. Un ejemplo podría ser la detección y clasificación del tipo del tumor. Además, se podría adaptar para otros contextos similares como puede ser el reconocimiento de anomalías en otras zonas del cuerpo como por ejemplo la detección de lunares cancerígenos. Otro posible uso sería su aplicación en animales, lo cual ayudaría mucho en las clínicas veterinarias.

7. Glosario

DT: Árboles de decisión.

IA: Inteligencia artificial.

KNN: Algoritmo de k-vecinos más cercanos.

RR.NN.AA: Redes de neuronas artificiales.

SVM: Maquina de soporte vectorial.

TC: Tomografía computerizada.

8. Bibliografía

[1] Javeria Amin y col. "Big data analysis for brain tumor detection: Deep convolutional neural networks". En: *Future Generation Computer Systems* 87 (2018), págs. 290-297.

[2] Ali Ari y Davut Hanbay. "Deep learning based brain tumor classification and detection system". En: *Turkish Journal of Electrical Engineering & Computer Sciences* 26.5 (2018), págs. 2275-2286.

[3] Hao Dong y col. *Automatic brain tumor detection and segmentation using U-Net based fully convolutional networks*. 2017, págs. 506-517.

[4] Todd C Hollon y col. "Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks". En: *Nature medicine* 26.1 (2020), págs. 52-58.

[5] A Jayachandran, N Zafar Ali Khan y Sudarson Rama Perumal. "Automatic Brain Cancer Classification in MRI Images Based On Ensemble Light Weight Deep Learning Network". En: (2022).

[6] Gabriel Moraes y col. "Image-Based Real-Time Path Generation Using Deep Neural Networks". En: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, págs. 1-8.

[7] Sidra Sajid, Saddam Hussain y Amna Sarwar. "Brain tumor detection and segmentation in MR images using deep learning". En: *Arabian Journal for Science and Engineering* 44.11 (2019), págs. 9249-9261.

[8] Dheiver Santos y Ewerton Santos. "Brain Tumor Detection Using Deep Learning". En: *medRxiv* (2022).

[9] Eugene Shkolyar y col. "Augmented bladder tumor detection using deep learning". En: *European urology* 76.6 (2019), págs. 714-718.