

Proyecto Fin de Máster

Máster en Ingeniería de Telecomunicación

Identificación temprana de crisis epilépticas
utilizando técnicas de aprendizaje automático

Autor: Andrés García-Baquero León

Tutor: Rubén Martín Clemente

Dep. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017



Proyecto Fin de Master
Master en Ingeniería de Telecomunicación

Identificación temprana de crisis epilépticas utilizando técnicas de aprendizaje automático

Autor:
Andrés García-Baquero León

Tutor:
Rubén Martín Clemente
Profesor titular

Dep. de Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2017

Proyecto Fin de Master: Identificación temprana de crisis epilépticas utilizando técnicas de aprendizaje automático

Autor: Andrés García-Baquero León

Tutor: Rubén Martín Clemente

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2017

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

Este trabajo fin de master significa el fin de mi época de estudiante como tal y sería imposible agradecer en estas líneas a todas las personas que han influido en mí en ese tiempo. Tantísimas cosas han cambiado desde el final del grado, y también con muchas dificultades, incluso para acceder al master que ahora termino. Los verdaderos artífices de todo esto no son otros que mis padres, Andres y Loreto, a los que le debo la vida y la persona que soy y que nunca podré agradecerles lo suficiente. También a mi hermana, Fátima, el mejor regalo que mis padres me dieron.

Me gustaría también agradecer a toda la gente que compartió esa vida paralela de un año que fue el segundo curso del master en Milán. Sobre todo agradecer la amistad a Manu, Kike y Chamorro por hacer de ese año algo inolvidable y que quedará para el recuerdo en aquel ático de Antonio Bordini.

La vida pasa y las cosas cambian, y yo solo quiero verla pasar junto a vosotros.

Gracias.

*Andrés García-Baquero León
Sevilla, 2017*

Resumen

En este proyecto se busca si es posible predecir un ataque epiléptico a partir de un encefalograma de un paciente. De este encefalograma se extrae información relativa a la señal, la cual se procesa tanto en bloques como en bandas de frecuencias y se estudian ciertas características. Esta información pasa por una Máquina de soporte vectorial, que primero aprende a diferenciar eventos y luego realiza predicciones. Así pues, gracias a la máquina de soporte vectorial se puede ver si estas características estudiadas presentan alguna particularidad en alguna banda de frecuencia concreta que signifique la llegada de un ataque epiléptico.

Abstract

The aim of this project is to seek if it is possible to predict an epileptic crisis from an encephalogram of a patient. Information about the signal is extracted from this encephalogram, which is processed in blocks as well as in frequency bands in which certain characteristics are studied. This information passes through a Vector Support Machine, which first learns to differentiate events and then makes predictions. Thus, thanks to the vector support machine, it is possible to see if these studied characteristics have any particularity in any specific frequency band that means the arrival of an epileptic crisis.

Índice

Agradecimientos	vii
Resumen.....	ix
Abstract	xi
Índice.....	xii
Índice de Tablas.....	xiv
Índice de Figuras.....	xv
1 Introducción	17
1.1 Organización del proyecto	17
1.2 Base de datos	18
2 Señal del encefalograma	19
2.1 Introducción	19
2.2 Cerebro	19
2.3 Actividad eléctrica cerebral	21
2.4 El espectro del EEG.....	21
2.4.1 Banda delta δ	22
2.4.2 Banda theta θ	22
2.4.3 Banda alfa α	23
2.4.4 Banda mu μ	23
2.4.5 Banda beta β	23
2.4.6 Banda gamma γ	23
2.5 Encefalografía	23
2.6 Epilepsia	26
2.6.1 Algoritmos y aplicaciones de detección de crisis.....	26
2.6.2 Dificultad de la detección de crisis epilépticas.....	26
3 Selección de características	29
3.1 Introducción	29
3.2 Coeficiente de asimetría	30
3.2.1 Definición	30
3.3 Curtosis.....	30
3.3.1 Definición	30
3.4 Potencia.....	31
3.5 Filtrado	31
3.5.1 Descripción del filtro de Chevyshev	31
4 Máquina de soporte vectorial	33
4.1 Márgenes	33
4.2 Notación.....	34
4.3 Margen funcional y geométrico	34
4.4 Margén clasificador óptimo	36
4.5 Dualidad de Lagrange.....	37
4.6 Clasificadores de márgenes óptimos	38
4.7 Kernels	40

4.8	<i>Regularización y el caso no separable.....</i>	42
4.9	<i>Algoritmo SMO</i>	44
4.9.1	<i>Coordenadas</i>	44
4.9.2	<i>SMO</i>	45
4.10	<i>Aplicación Matlab para MSV.....</i>	47
5	Pruebas	51
5.1	<i>Paciente 1.....</i>	51
5.1.1	<i>Registro 1.....</i>	51
5.1.2	<i>Registro 2.....</i>	52
5.1.3	<i>Registro 3.....</i>	53
5.1.4	<i>Registro 4.....</i>	54
5.1.5	<i>Registro 5.....</i>	55
5.2	<i>Paciente 2.....</i>	56
5.2.1	<i>Registro 1.....</i>	56
5.2.2	<i>Registro 2.....</i>	57
5.2.3	<i>Registro 3.....</i>	58
5.2.4	<i>Registro 4.....</i>	58
5.2.5	<i>Registro 5.....</i>	59
5.3	<i>Paciente 3.....</i>	60
5.3.1	<i>Registro 1.....</i>	60
5.3.2	<i>Registro 2.....</i>	61
5.3.3	<i>Registro 3.....</i>	62
5.3.4	<i>Registro 4.....</i>	63
5.3.5	<i>Registro 5.....</i>	64
5.4	<i>Paciente 4.....</i>	65
5.4.1	<i>Registro 1.....</i>	65
5.4.2	<i>Registro 2.....</i>	66
5.4.3	<i>Registro 3.....</i>	67
5.4.4	<i>Registro 4.....</i>	67
5.4.5	<i>Registro 5.....</i>	68
5.5	<i>Paciente 5.....</i>	69
5.5.1	<i>Registro 1.....</i>	69
5.5.2	<i>Registro 2.....</i>	70
6	Resultados.....	72
8	Referencias	74
7	Anexos	76
7.1	<i>Script 1: bloques.m.....</i>	76
7.2	<i>Script 2: pruebas.m</i>	89
7.3	<i>Script 3: MSVc (kur.m)</i>	103
7.4	<i>Script 4: MSVa (asimetria.m).....</i>	105
7.5	<i>Script 5: MSVpa (pot_asin.m).....</i>	108
7.6	<i>Script 6: MSVpb (pot_bsin.m).....</i>	110
7.7	<i>Script 7: MSVpg (pot_gsin.m).....</i>	112

ÍNDICE DE TABLAS

Tabla 1 - Bandas espectrales usadas en EEG

22

ÍNDICE DE FIGURAS

Figura 1 - Morfología del cerebro humano	20
Figura 2 - Corteza cerebral	21
Figura 3 - Espectro de la señal EEG	22
Figura 4 - Técnicas de captura de la actividad cerebral	24
Figura 5 - Ejemplo de un encefalograma	25
Figura 6 - Partes de la neurona	25
Figura 7 - División en bloques	29
Figura 8 - ejemplo de margen	34
Figura 9 - Margen geométrico	35
Figura 10 - hiperplano separador con margen máximo	39
Figura 11 – Regularización y caso no separable	43
Figura 12 - Algoritmo de subida coordinada	44
Figura 13 - Restricciones algoritmo SMO	46

1 INTRODUCCIÓN

"Si tú eres un buen comandante, oblígame a luchar, aunque no quiera".

- Cayo Mario, militar y político romano, tercer fundador de Roma -

El ser humano siempre se ha preocupado por las situaciones impredecibles que acaecen en su alrededor. El objetivo de las grandes mentes de la antigüedad ha sido no otro que la búsqueda del porqué de estas situaciones y llegado el caso evitarlo, si es posible o minimizar sus consecuencias.

Desde las más visibles como los terremotos hasta las más concretas como las enfermedades. Una de estas últimas más visible e incomprensible al ser humano primitivo era la epilepsia.

Hasta que la ciencia no descubrió la electricidad y el estudio del cuerpo humano no fue considerado como ámbito científico no se vieron avances en esta faceta. Además, el desconocimiento que ha habido sobre la epilepsia ha hecho que siempre sea visto de forma recelosa.

Mucho se ha avanzado desde entonces afortunadamente.

En este proyecto trato de añadir un pequeño grano de arena a este tema desde un punto de vista tecnológico, usando técnicas de procesado de señal.

1.1 Organización del proyecto

La división del proyecto será la siguiente:

- Señal del encefalograma:

En este capítulo se describe la señal que produce el encefalograma, información general sobre la epilepsia y el funcionamiento eléctrico del cerebro.

- Selección de características

En este capítulo se describen las características escogidas del encefalograma.

- Máquina de soporte vectorial

En este capítulo se describe la máquina de soporte vectorial desde el punto de vista teórico, además de la aplicación Matlab que la realiza.

- Pruebas

En este capítulo se describen las pruebas realizadas para comprobar el funcionamiento.

- Resultados

En este capítulo se comentan los resultados de las pruebas realizadas.

1.2 Base de datos

Esta base de datos ha sido recogida del Hospital Infantil de Boston y consiste en grabaciones de EEG de sujetos pediátricos con crisis intratables. Los sujetos fueron monitorizados hasta varios días después de la retirada de la medicación anti-convulsión con el fin de caracterizar sus crisis y evaluar la necesidad de la intervención quirúrgica.

Se recolectaron grabaciones agrupadas en 23 casos, de 22 sujetos (5 varones, edades 3-22 y 17 mujeres, edades 1,5-19).

Cada caso (chb01, chb02, etc.) contiene entre 9 y 42 archivos continuos de un solo sujeto. Las limitaciones de hardware resultaron en espacios entre archivos numerados consecutivamente, durante los cuales las señales no fueron registradas; en la mayoría de los casos, los vacíos son de 10 segundos o menos, pero ocasionalmente hay vacíos mucho más largos. Con el fin de proteger la privacidad de los sujetos, toda la información médica protegida (PHI) en los archivos originales ha sido reemplazada con información sustituta en los archivos proporcionados aquí. Las fechas en los archivos originales han sido reemplazadas por fechas sustitutivas, pero las relaciones de tiempo entre los archivos individuales pertenecientes a cada caso se han conservado. En la mayoría de los casos, los archivos contienen exactamente una hora de señales EEG digitalizadas, aunque las pertenecientes al caso chb10 tienen una duración de dos horas y las correspondientes a los casos chb04, chb06, chb07, chb09 y chb23 tienen una duración de cuatro horas; ocasionalmente, los archivos en los que se registran las incautaciones son más cortos.

Todas las señales se muestran a 256 muestras por segundo con una resolución de 16 bits. La mayoría de los archivos contienen 23 señales EEG (24 o 26 en algunos casos). Para estas grabaciones se utilizó el sistema internacional 10-20 posiciones de electrodos en el EEG. En algunos registros, también se registran otras señales, como una señal de ECG en los últimos 36 archivos pertenecientes al caso chb04 y una señal de estímulo del nervio vagal (VNS) en los últimos 18 archivos pertenecientes al caso chb09. En algunos casos, se intercalaron hasta 5 señales "falsas" (denominadas "-") entre las señales EEG para obtener un formato de visualización fácil de leer; estas señales ficticias pueden ser ignoradas.

En este proyecto se han usado los sujetos chb01, chb02, chb03, chb05 y chb06. Los archivos de cada sujeto se han seleccionado aleatoriamente.

2 SEÑAL DEL ENCEFALOGRAMA

"No basta adquirir sabiduría, es preciso además saber usarla"

-Cicerón-

En este capítulo se presenta una breve introducción a las señales del electroencefalograma (EEG). En primer lugar, se introducen conceptos generales sobre el cerebro, para luego realizar una descripción de las señales de EEG y la encefalografía.

Se presenta también una breve descripción de las señales de EEG de enfermos con epilepsia, y sus características particulares durante la crisis.

Finalmente se presentan las señales de EEG que se utilizarán a lo largo de la tesis para aplicar las técnicas de procesamiento estudiadas, haciendo una breve descripción de sus características.

2.1 Introducción

El electroencefalograma (EEG) es el registro de la actividad eléctrica de las neuronas del encéfalo. Dicho registro posee formas muy complejas que varían mucho con la localización de los electrodos y entre individuos, más adelante explicaré como se determina la localización de los electrodos. Esto es debido al gran número de interconexiones que presentan las neuronas y por la estructura no uniforme del encéfalo.

Cabe añadir que el electroencefalograma es una técnica diagnóstica fácil de realizar, barata y no es invasiva.

2.2 Cerebro

El encéfalo es un órgano esencial del cuerpo humano, controla funciones tales como la respiración, la vista, el tacto, el movimiento, la temperatura, y todos los procesos que regulan nuestro cuerpo. Está contenido en el cráneo y se compone de tres partes bien diferenciadas: el tronco cerebral, el cerebelo y el cerebro, cuya superficie externa es conocida como corteza cerebral (Figura 1).

La corteza cerebral presenta un conjunto de prominencias: circunvoluciones y cisuras. La forma y la ubicación de las mismas pueden variar de una persona a otra; sin embargo son utilizadas para delimitar las diferentes zonas o regiones del cerebro. La más importante es la denominada cisura longitudinal, que divide al cerebro en los hemisferios derecho e izquierdo. A su vez, cada hemisferio se subdivide en cuatro lóbulos: el frontal, el parietal, el temporal y el occipital (Figura 1).

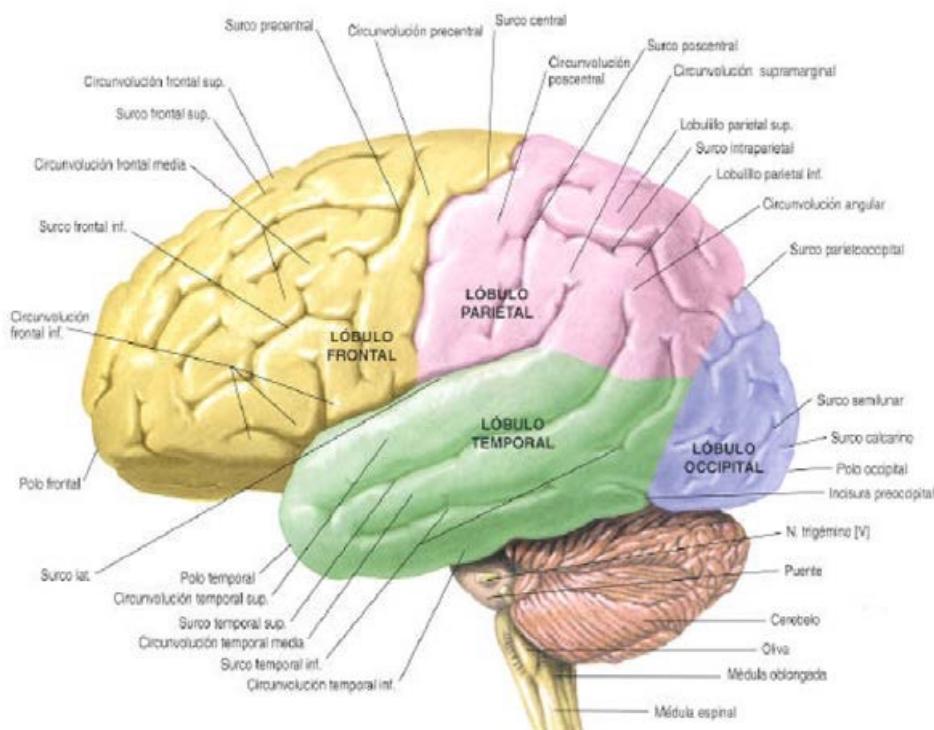


Figura 1 - Morfología del cerebro humano

Cada una de las áreas de la corteza cerebral es responsable de funciones específicas. Así por ejemplo, los lóbulos parietales contienen un detallado mapa de la sensibilidad, los lóbulos occitales se encargan de la visión, los temporales del oído, y los lóbulos frontales son los encargados de un gran número de funciones, como resolver problemas complejos o controlar la actividad muscular.

En particular, el área responsable de los movimientos voluntarios de los músculos del cuerpo se localiza delante de la cisura Central, y se le conoce como corteza motora primaria, mientras que por detrás de dicha cisura se encuentra la corteza sensorial primaria, responsable del análisis y percepción del sentido del tacto (Figura 2).

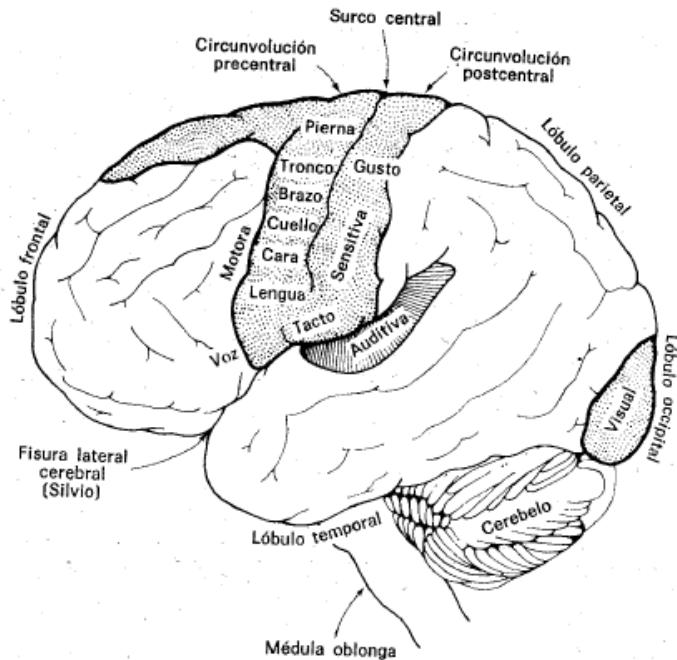


Figura 2 - Corteza cerebral

2.3 Actividad eléctrica cerebral

La actividad eléctrica del cerebro está formada por biopotenciales, que se originan en la membrana externa de las células excitables, tales como las que componen el tejido nervioso o muscular. Estas señales eléctricas son de naturaleza iónica.

La frecuencia de la actividad eléctrica espontánea del cerebro, refleja tanto las propiedades intrínsecas de las membranas de una neurona, como la organización e interconectividad de la red a la que pertenecen. Estas redes pueden ser del tipo local, y estar formadas por neuronas de una región de la corteza cerebral, o bien estar distribuida por diferentes partes del cerebro, abarcando un gran número de neuronas. La actividad sincrónica de un gran número de neuronas produce señales eléctricas de mayor amplitud y baja frecuencia, como por ejemplo el ritmo cerebral alfa. Por el contrario, cuando el numero de neuronas sincronizadas es menor y se encuentran en una región más localizada, se manifiestan a mayores frecuencias y con menores amplitudes, como los ritmos beta o gamma.

Esta actividad eléctrica se propaga a través del tejido circundante y se detecta con los electrodos que cumplen la función de transductores, convirtiendo las corrientes iónicas en corrientes electrónicas, para su posterior procesamiento.

2.4 El espectro del EEG

El espectro de frecuencias de las señales electroencefalográficas se extiende desde algunas décimas de Hz hasta aproximadamente 100 Hz. En el análisispectral de señales de EEG, se definen ciertas bandas de importancia clínica que se denominan con las letras griegas δ , θ , α , β y γ . Sin embargo con el paso del tiempo se han ido descubriendo nuevos ritmos que en algunos casos comparten estas bandas de frecuencias, y se diferencian en características como localización o funciones del cerebro.

Si bien estas bandas no tienen límites precisos abarcan aproximadamente los siguientes intervalos:

δ	< 4 Hz
θ	4-8 Hz
α	8-12 Hz
β	12-32 Hz
γ	> 32 Hz
μ	8-13 Hz

Tabla 1 - Bandas espectrales usadas en EEG

Si bien en aplicaciones clínicas se utilizan fundamentalmente la zona de bajas frecuencias, actualmente también son motivo de interés las componentes de alta frecuencia, especialmente alrededor de 40 Hz.

La distribución espectral de la energía de las señales de EEG depende de la actividad mental en ejecución. En la Figura 3 se muestran espectros típicos de EEG. Corresponden a registros tomados con electrodos superficiales en la zona occipital con ojos abiertos y con ojos cerrados sobre un ancho de banda de 32 Hz. Se puede observar, en ambos casos, una predominancia de la banda alfa.

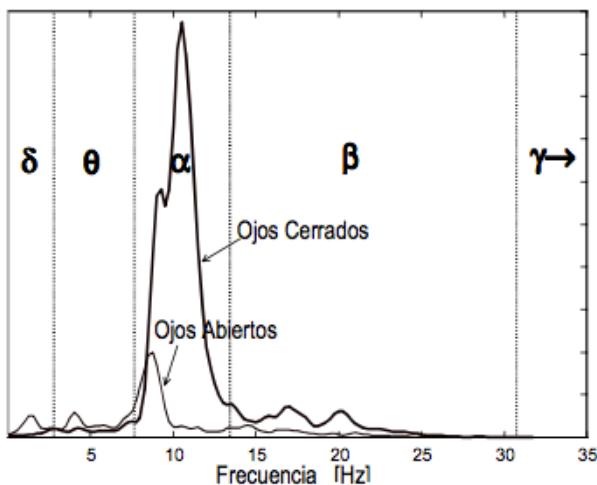


Figura 3 - Espectro de la señal EEG

2.4.1 Banda delta δ

Es un ritmo de gran amplitud y baja frecuencia. Se encuentran típicamente entre 0,5 y 3,5 Hz y presenta amplitudes de 20 a 200 μ V. Se encuentra en individuos adultos sanos, exclusivamente durante el sueño profundo. En caso de detectarse en una persona despierta, puede indicar que existe algún tipo de anormalidad en el cerebro.

2.4.2 Banda theta θ

Este ritmo es en general menos común que los demás. Se presenta en la banda de 4 a 7 Hz, con amplitudes que oscilan entre 20 y 100 μ V. Se encuentra presente con mayor frecuencia en niños. En adultos sanos, se pueden detectar en estado de adormecimiento y sueño. Se registra en el lóbulo

temporal.

2.4.3 Banda alfa α

El ritmo alfa se manifiesta principalmente en la banda de frecuencias de 8 a 13 Hz, con amplitudes que oscilan entre 20 y 60 μ V. Se encuentran en el electroencefalograma de la mayoría de los adultos sanos, con los ojos cerrados o con reposo visual, despiertos con un estado mental tranquilo y de reposo. El ritmo alfa es bloqueado o atenuado por la atención, especialmente visual y esfuerzo mental o físico. Durante el sueño profundo también desaparecen las ondas alfa. Se observa principalmente en la zona posterior de la cabeza, en el área occipital, parietal y la región temporal posterior.

2.4.4 Banda mu μ

Se manifiesta en la banda de 8 a 13 Hz y su amplitud es menor a 50 μ V. Si bien sus características de frecuencia y amplitud son similares a los del ritmo α , presenta características topográficas y fisiológicas claramente diferentes. El ritmo μ se detecta en la corteza motora primaria, bloqueándose con la realización de movimientos o estímulos táctiles y visuales; e incluso con la imaginación o preparación de un movimiento.

2.4.5 Banda beta β

Es un ritmo irregular, con frecuencias entre 13 y 30 Hz. Su amplitud aproximada está entre 2 y 20 μ V. Suele asociarse a un estado de concentración mental. Se detecta principalmente en la región central y frontal del cuero cabelludo, cerca o sobre la corteza motora primaria. Son comunes cuando la persona está envuelta en actividad mental o física. La banda central de este ritmo está relacionada con el movimiento de las extremidades, tomando sus valores de amplitud máximos algunas centésimas de segundo luego de la realización de un movimiento.

2.4.6 Banda gamma γ

Este ritmo se manifiesta a frecuencias mayores a los 30 Hz y amplitudes entre 5 y 10 μ V. Es una actividad armónica que se presenta como respuesta a estímulos sensoriales, como sonidos contundentes o luces intermitentes. Esta actividad se puede observar en una zona extensa de la corteza cerebral, manifestándose principalmente en la zona frontal y la central.

2.5 Encefalografía

Primeramente, resulta interesante conocer por qué se suele elegir el electroencefalograma como método de medida de la actividad cerebral. Teniendo en cuenta la Figura 4, que muestra una clasificación de las diferentes técnicas de captura de la actividad cerebral en función de la resolución espacio-temporal que requieren, se observa cómo el EEG es de los menos invasivos. Aparte, ofrece las mejores prestaciones en relación calidad-precio y su fabricación es sencilla.

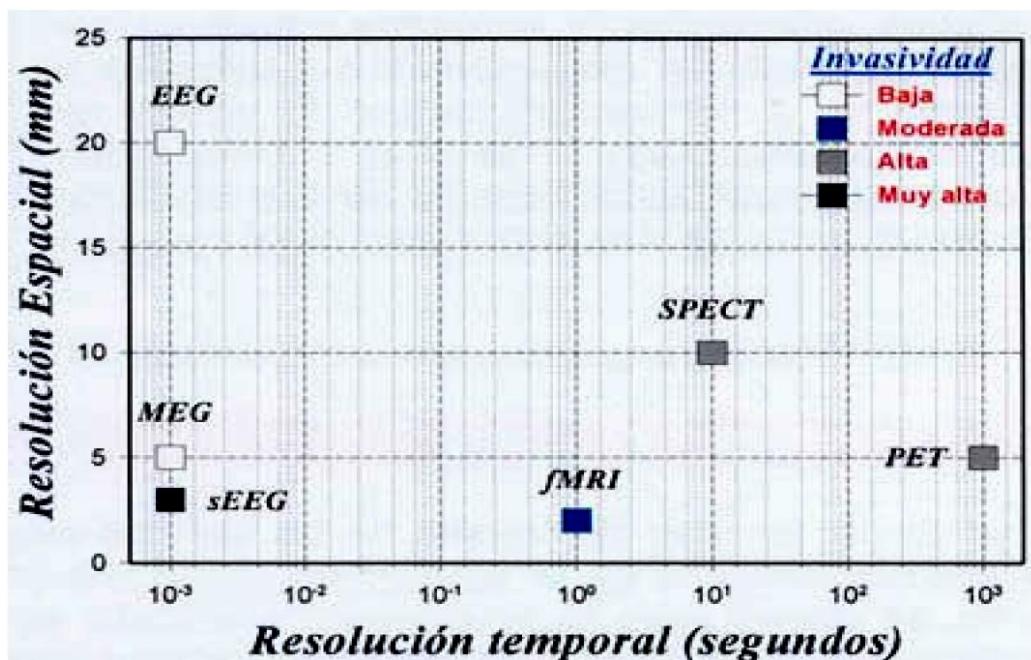


Figura 4 - Técnicas de captura de la actividad cerebral

La electroencefalografía es uno de los métodos electro-fisiológicos más antiguos de la ciencia moderna, datando de 1924 por Hans Berger. En 1929, él mismo fue capaz de constatar la existencia de los ritmos alfa y beta en sus análisis de los EEGs. El EEG está basado en las corrientes de naturaleza iónica existentes en la corteza cerebral, que son resultado de la actividad cerebral y pueden ser capturadas con unos electrodos colocados en el exterior del cráneo. Anteriormente, estas corrientes iónicas tienen que ser convertidas a eléctricas, condición necesaria para que los electrodos metálicos puedan transportar la corriente hasta el amplificador de instrumentación. Para maximizar la eficiencia de la transducción iónica a eléctrica, normalmente se usa un electrolito en contacto con la piel y un electrodo de oro, plata o algún derivado químico. La señal eléctrica recogida, se amplifica y se representa en formas de ondas a lo largo del tiempo (ver Figura 5).

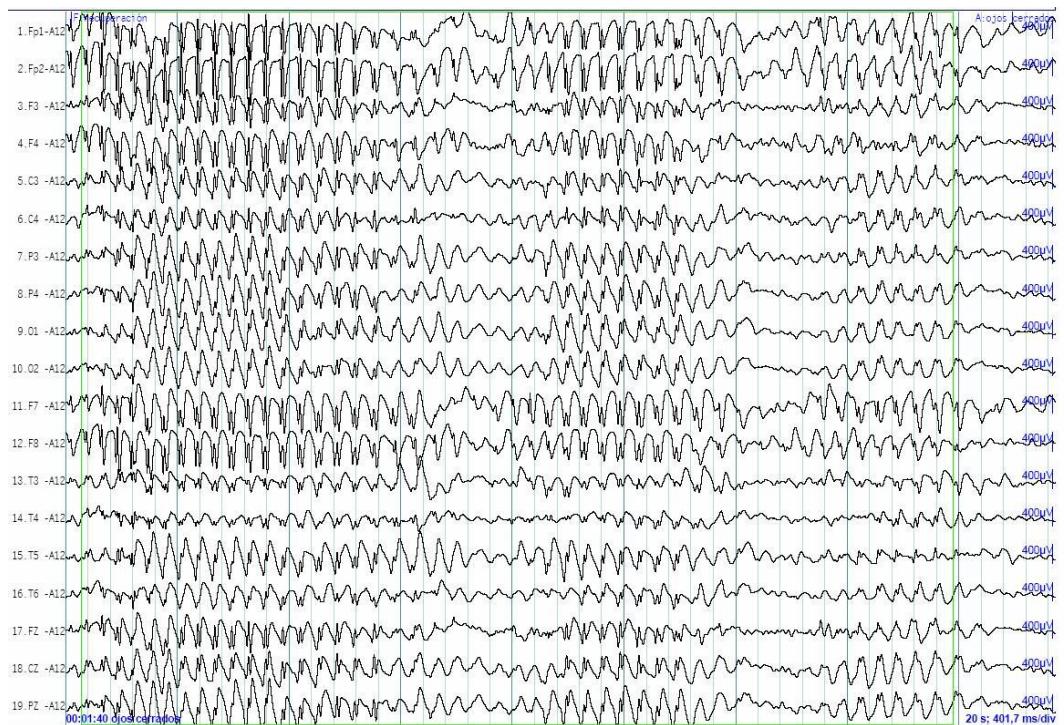


Figura 5 - Ejemplo de un encefalograma

El EEG se encarga de leer el sistema nervioso y a partir de él, se analiza la información. El sistema nervioso es un conjunto de tejidos dentro de nuestro cuerpo, encargados de captar y procesar rápidamente las señales internas y externas, tomando el control y coordinación sobre los demás órganos, para así, lograr una oportuna y eficaz interacción con el medio ambiente cambiante. La unidad básica del sistema nervioso es la neurona, la cual tiene la capacidad de comunicarse eléctricamente con otras células, sean éstas nerviosas o no. La información viaja entre neuronas por medio de impulsos eléctricos que se transmiten de unas neuronas a otras. Estos impulsos, se reciben de otras neuronas en las dendritas y pasan a través de la neurona hasta ser conducidas por el axón a los terminales de salida, los cuales pueden conectarse con otra neurona, fibras musculares o glándulas (ver Figura 6).

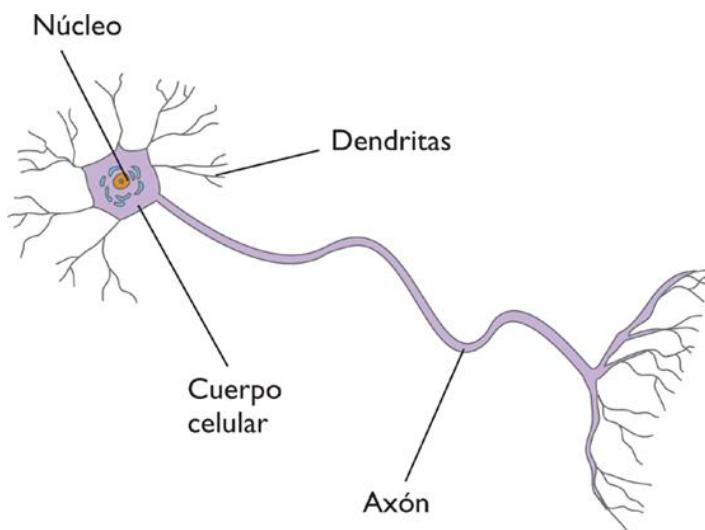


Figura 6 - Partes de la neurona

Cuando se produce un estímulo externo, el sistema nervioso actúa de la siguiente manera: El estímulo es recibido en alguna región sensorial capturando alguna información, la cual es transportada por el sistema nervioso (a través de las neuronas) hasta una componente integradora en donde se analiza. Esta componente elabora la respuesta, que es conducida a través de las neuronas hacia fibras musculares (respuesta motora) o hacia glándulas (secreción glandular). Hay que tener en cuenta que la actividad cerebral es producida por un número muy elevado de neuronas (aproximadamente cien mil millones en un cuerpo humano medio) y cada una de las tareas que nuestro cuerpo puede realizar, provoca una actividad cerebral con forma e intensidad diferentes, además de zonas del sistema nervioso.

2.6 Epilepsia

La epilepsia es un trastorno crónico del sistema nervioso central que predispone al individuo a experimentar crisis recurrentes. Una crisis es una repentina y transitoria distorsión de la actividad eléctrica del cerebro que produce ciertos síntomas. El rango de estos síntomas pueden variar desde un lapsus en la atención, alucinaciones sensoriales, o incluso una convulsión general del cuerpo.

La epilepsia no es un único trastorno, sino una familia de síndromes que comparten características en las crisis. Este trastorno puede resultar de una mutación heredada en un mecanismo molecular que regula el comportamiento de la neurona, la migración o la organización. Alternativamente, puede desarrollarse como resultado del trauma cerebral como un duro golpe en la cabeza, un golpe, una infección cerebral o una neoplasia de cerebro.

Cincuenta millones personas por todo el mundo están diagnosticadas de epilepsia. En los Estados Unidos la epilepsia afecta a 3 millones de personas y es el tercer desorden neurológico más común después del Alzheimer y el ataque cerebrovascular. En España los diagnosticados son unos 400.000.

2.6.1 Algoritmos y aplicaciones de detección de crisis

Un detector de crisis puede clasificarse como un detector de aparición de convulsiones o un detector de eventos de crisis. El propósito de un detector de aparición de convulsiones es reconocer que una convulsión ha comenzado con la menor tardanza posible, pero no necesariamente con la mayor precisión posible. Por el contrario, el propósito de un detector de eventos de convulsiones es identificar convulsiones con la mayor precisión posible, pero no necesariamente con la menor tardanza. Los detectores de aparición de convulsiones son adecuados para aplicaciones que requieren una respuesta rápida a una convulsión, mientras que los detectores de eventos de convulsiones son adecuados para aplicaciones que requieren una buena tasa de precisión en un período de tiempo.

2.6.2 Dificultad de la detección de crisis epilépticas

La detección de aparición de convulsiones y la detección de eventos de crisis se obtienen normalmente mediante el análisis del electroencefalograma (EEG). El EEG se mide utilizando electrodos no invasivos. Cuando éstos están dispuestos en el cuero cabelludo de un individuo, se conoce como 'EEG en el cuero cabelludo'; y cuando se mide con electrodos colocados en la superficie del cerebro o dentro de sus profundidades se denomina 'EEG intracranegal'.

La propiedad del EEG en el cuero cabelludo y EEG intracranegal que más complica la tarea de detección es su variabilidad en las personas con epilepsia, tanto en los estados sin crisis como con crisis. Por lo general, el inicio de una convulsión, un conjunto de canales del EEG desarrolla actividad rítmica que refleja la sincronía neuronal subyacente. Tanto la ubicación de los canales de EEG involucrados, así como el contenido espectral de la actividad rítmica varía según los individuos. Además, la forma del EEG durante una crisis de un paciente puede que se asemeje la forma de un EEG normal, sin convulsiones.

En el EEG en el cuero cabelludo la tarea de detección de crisis se complica aún más por las propiedades físicas de la señal. En el EEG en el cuero cabelludo es más sensible a la actividad de las neuronas en la superficie del cerebro; en consecuencia, la actividad de neuronas de las estructuras

cerebrales profundas casi no influye en el EEG en el cuero cabelludo. Cuando la red neuronal epiléptica es profunda dentro del cerebro, el EEG en el cuero cabelludo puede reflejar secuelas físicas de la convulsión, como el parpadeo repetitivo del ojo o contracciones musculares, que reflejan actividad neuronal sincrónica. Las crisis de este tipo son difíciles de detectar con alta precisión y baja latencia en actividades tales como parpadeo de ojos y las contracciones musculares que se observan rutinariamente en un individuo diariamente.

Otra propiedad de EEG en el cuero cabelludo que hace difícil la detección de crisis es su susceptibilidad a la contaminación por fuentes no fisiológicas. La influencia de los cables de electrodos, alteraciones en la interfaz electrodo-piel y el acoplamiento de los armónicos de AC de la maquinaria eléctrica pueden producir cambios espectrales que afectan el funcionamiento de un detector de crisis.

3 SELECCIÓN DE CARACTERÍSTICAS

"La historia ofrece el medio mejor de preparación para los que han de tomar parte en los asuntos públicos."

- Polibio, Historiador griego -

En este capítulo se analizarán las características que se han obtenido a partir de los archivos de la base de datos descrita en el capítulo uno. El archivo está compuesto por 23 señales distintas, cada una con la información de un electrodo diferente.

3.1 Introducción

Se han definido primeramente unos filtros de Chevyshev para separar la matriz de datos en las bandas de frecuencias descritas en el capítulo anterior.

Después, los datos se han procesado en forma de bloques, cada bloque de manera totalmente independiente al siguiente. La división del archivo en bloques se ha hecho para extraer las características de cada bloque. Esto servirá para ver la evolución del paciente a lo largo del tiempo y así ver si es posible prever el ataque. Se ha dejado una ventana de superposición modificable para estudiar el comportamiento de valores próximos al de la división del bloque (línea azulada en la figura).

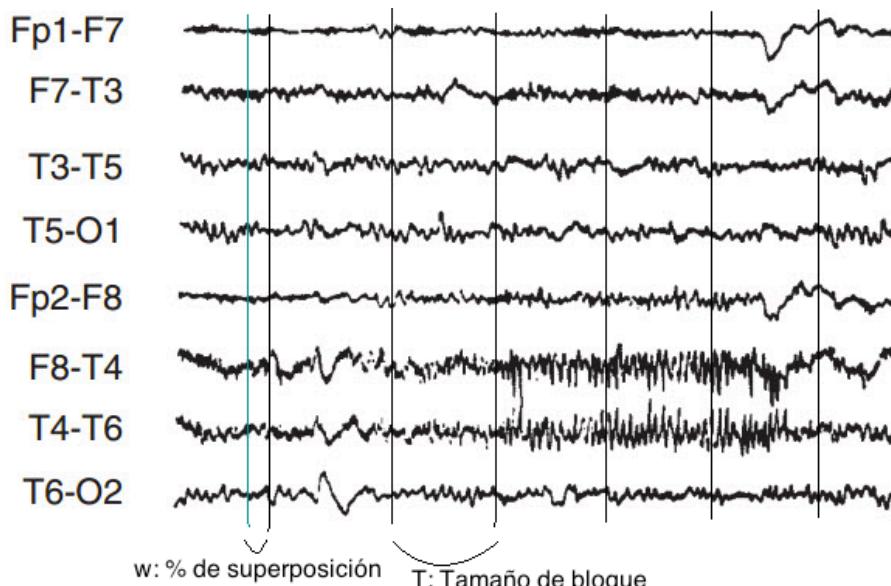


Figura 7 - División en bloques

La matriz de cada característica procesada ha sido creada a partir de la información propia de cada bloque. Es decir, la matriz de la media por ejemplo será de 23 filas por N columnas. Siendo 23 el número de señales del encefalograma y N el número de bloques en el cual se han dividido los datos.

Las características que se han estudiado son las siguientes:

3.2 Coeficiente de asimetría

Se ha calculado el coeficiente de asimetría de cada señal del encefalograma en cada bloque de medición de estudio para cada banda de frecuencia. En este caso tenemos una matriz de 23 por N, siendo N el número de bloques. Se ha calculado con la función Matlab *skewness*. Estos valores han sido introducidos en la matriz *fisher*, para cada banda de frecuencia.

3.2.1 Definición

Las medidas de asimetría son indicadores que permiten establecer el *grado de simetría* (o asimetría) que presenta una distribución de probabilidad de una variable aleatoria sin tener que hacer su representación gráfica. Como eje de simetría consideramos una recta paralela al eje de ordenadas que pasa por la media de la distribución. Si una distribución es simétrica, existe el mismo número de valores a la derecha que a la izquierda de la media, por tanto, el mismo número de desviaciones con signo positivo que con signo negativo. Decimos que hay asimetría positiva (o a la derecha) si la "cola" a la derecha de la media es más larga que la de la izquierda, es decir, si hay valores más separados de la media a la derecha. Diremos que hay asimetría negativa (o a la izquierda) si la "cola" a la izquierda de la media es más larga que la de la derecha, es decir, si hay valores más separados de la media a la izquierda.

La medida de asimetría más utilizada parte del uso del tercer momento estándar. La razón de esto es que nos interesa mantener el signo de las desviaciones con respecto a la media, para obtener si son mayores las que ocurren a la derecha de la media que las de la izquierda.

El coeficiente de asimetría de Fisher, representado por γ_1 , se define como:

$$\gamma_1 = \frac{\mu_3}{\sigma^3}$$

donde μ_3 es el tercer momento en torno a la media y σ es la desviación estándar.

Si $\gamma_1 > 0$, la distribución es asimétrica positiva o a la derecha.

Si $\gamma_1 < 0$, la distribución es asimétrica negativa o a la izquierda.

Si la distribución es simétrica, entonces sabemos que $\gamma_1 = 0$. El recíproco no es cierto: es un error común asegurar que si $\gamma_1 = 0$ entonces la distribución es simétrica (lo cual es falso).

3.3 Curtosis

Se ha calculado el coeficiente de curtosis de cada señal del encefalograma en cada bloque de medición de estudio para cada banda de frecuencia. En este caso tenemos una matriz de 23 por N, siendo N el número de bloques. Se ha calculado con la función Matlab *kurtosis*. Estos valores han sido introducidos en la matriz *curtosis*, para cada banda.

3.3.1 Definición

La medida de curtosis trata de estudiar la proporción de la varianza que se explica por la combinación de datos extremos respecto a la media en contraposición con datos poco alejados de la

misma.

Una mayor curtosis implica una mayor concentración de datos muy cerca de la media de la distribución coexistiendo al mismo tiempo con una relativamente elevada frecuencia de datos muy alejados de la misma. Esto explica una forma de la distribución de frecuencias con colas muy elevadas y con un centro muy apuntado.

Un coeficiente de apuntamiento o de curtosis es el basado en el cuarto momento con respecto a la media y se define como:

$$\beta_2 = \frac{\mu_4}{\sigma^4}$$

donde μ_4 es el 4º momento centrado o con respecto a la media y σ es la desviación estándar.

3.4 Potencia

Se ha calculado la potencia de cada señal del encefalograma en cada bloque de medición de estudio para cada banda de frecuencia. En este caso tenemos una matriz de 23 por N, siendo N el número de bloques. Se ha calculado con la función Matlab *obw*. Estos valores han sido introducidos en la matriz *potencia*, para cada banda. Al ser varianza y potencia redundantes y ser un cálculo importante, se ha calculado como comprobación de ésta por otro método.

3.5 Filtrado

Como se ha comentado en la introducción, se han definido unos filtros de Chevyshev para separar la matriz de datos en las bandas de frecuencia más relevantes.

3.5.1 Descripción del filtro de Chevyshev

Con los filtros de Chebyshev se consigue una caída de la respuesta en frecuencia más pronunciada en frecuencias bajas debido a que permiten rizado en alguna de sus bandas (paso o rechazo). A diferencia del Filtro de Butterworth donde los polos se distribuyen sobre una circunferencia, los polos del filtro Chebyshev lo hacen sobre una elipse; sus ceros se encuentran en el eje imaginario.

El tipo de filtro usado es el filtro de Chevyshev de tipo I que son filtros que únicamente tienen polos, presentan un rizado constante en la banda pasante y presentan una caída monótona en la banda de rechazo. La respuesta en frecuencia es:

$$|H(\Omega)|^2 = \frac{1}{1 + \varepsilon^2 T_n^2(\frac{\Omega}{\Omega_c})} \text{ para } 0 < \varepsilon \leq 1$$

donde N es el orden del filtro, Ω_c es la frecuencia de corte, Ω es la frecuencia analógica compleja ($\Omega_c = jw$) y $T_N(x)$ es el polinomio de Chebyshev de orden N, que se define como: $T_{N+1}(x) = 2 * x * T_N(x) - T_{N-1}(x)$ con $T_0(x) = 1$ y $T_1(x) = x$.

estos filtros la frecuencia de corte no depende de N y el módulo de su respuesta en frecuencia oscila (rizado) entre 1 y $\frac{1}{\sqrt{1+\varepsilon^2}}$.

4 MÁQUINA DE SOPORTE VECTORIAL

“Ten presente que los hombres, hagas lo que hagas, siempre serán los mismos.”

- Marco Aurelio, Emperador Romano -

En este apartado se presenta el algoritmo de funcionamiento de la máquina de soporte vectorial (SVM). Para tener una idea general de SVM, primero se deberá hablar de los conceptos de margen y “hueco” dentro de la idea de separar datos. Luego, se hablará del clasificador de margen óptimo, donde se introducirá la digresión sobre la dualidad de Lagrange. También se verá la aplicación en kernels, que dan un buen ejemplo de aplicación eficiente de SVMs y, por último, se comentará el algoritmo SMO, que da una implementación eficiente de SVMs.

4.1 Márgenes

Considere la regresión logística, donde la probabilidad $p(y = 1 | x; \theta)$ es modelada por $h_\theta(x) = g(\theta^T x)$. Entonces predeciríamos “1” en una entrada x si y sólo si $h_\theta(x) \geq 0,5$, o equivalentemente, si y sólo si $\theta^T x \geq 0$. Considere un ejemplo positivo ($y = 1$). Cuanto mayor sea $\theta^T x$, más grande se vuelve $h_\theta(x) = p(y = 1 | x; w, b)$, y por lo tanto también es mayor nuestro grado de confianza. Así, se puede pensar que nuestra predicción de que $y = 1$ es más probable cuando $\theta^T x \gg 0$. Similarmente, pensando esto se podría afirmar que $y = 0$ si $\theta^T x \ll 0$. Dado un conjunto de ejemplo, de nuevo parece que se habrían encontrado un buen ajuste para los datos del ejemplo si se puede encontrar θ para que $\theta^T x^{(i)} \gg 0$ cuando $y^{(i)} = 1$ y $\theta^T x^{(i)} \ll 0$ cuando $y^{(i)} = 1$, ya que esto reflejaría una posición muy fiable para cualquier ejemplo de entrenamiento.

Para una descripción visual diferente, considere la siguiente figura 8, en la que x representa ejemplos positivos, y denota ejemplos negativos, el límite de decisión (la línea dada por la ecuación $\theta^T x = 0$, también llamado el hiperplano de separación), además de tres puntos etiquetados como A, B y C.

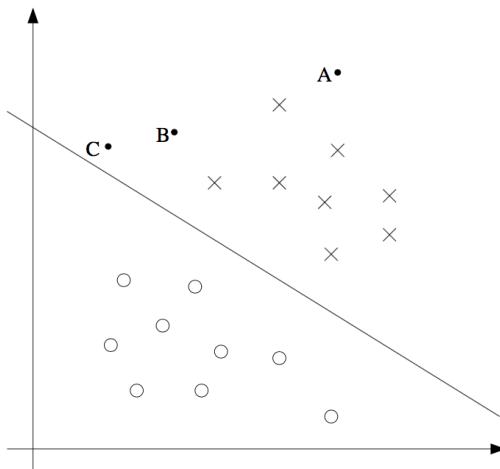


Figura 8 - ejemplo de margen

Se observa que el punto A está muy lejos del límite de decisión. Si se nos pide hacer una predicción para el valor del punto A en la coordenada y , parece ser que muy probablemente que $y = 1$. Por el contrario, el punto C está muy cerca del límite de decisión, parece lógico pensar que mientras no haya cambios el valor de y sea 1, sin embargo un ligero cambio en su posición nos incitaría a cambiar nuestra predicción e y valdría 0. Por lo tanto, tenemos mucha más confianza en nuestra predicción en A que en C. El punto B se encuentra entre estos dos casos, y más ampliamente, vemos que si un punto está muy lejos del hiperplano de separación, entonces podemos estar significativamente más seguros en las predicciones. Una vez más, se podría pensar que sería bueno si dado un conjunto de ejemplos, logramos encontrar un límite de decisión que nos permita hacer predicciones con mucha seguridad (lejos del límite de la decisión).

4.2 Notación

Para facilitar nuestra discusión sobre las SVM, primero debemos introducir una nueva notación para hablar de su clasificación. Se considerará un clasificador lineal para el problema de clasificación binaria con etiquetas y y características x . A partir de ahora, usaremos $y \in \{-1, 1\}$ (en lugar de $\{0, 1\}$) para denotar las etiquetas de clase. Además, en lugar de parametrizar nuestro clasificador lineal con el vector θ , se utilizará los parámetros w, b , y nuestro clasificador será como

$$h_{w,b}(x) = g(w^T x + b)$$

Aquí, $g(z) = 1$ si $z \geq 0$, y $g(z) = -1$ de lo contrario. Esta notación con w y b nos permite tratar explícitamente el término de intercepción b de forma separada de los otros parámetros. Así, b toma el papel de θ_0 , y w toma el papel de $[\theta_1 \dots \theta_n]^T$. Obsérvese también que, a partir de nuestra definición de g anterior, nuestro clasificador predecirá directamente 1 o -1 sin pasar por la etapa intermedia de estimar la probabilidad de que y sea 1.

4.3 Margen funcional y geométrico

Formalicemos la notación para los márgenes funcionales y geométricos. Dado un conjunto de ejemplo $(x^{(i)}, y^{(i)})$, definimos el margen funcional de (w, b) con respecto al ejemplo

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x + b).$$

Nótese que si $y^{(i)}=1$ entonces el margen funcional será grande, y por lo tanto nuestra predicción será fiable. Se necesitará por tanto un término $(w^T x + b)$ grande y positivo. Por el contrario, si $y^{(i)}=-1$, entonces el margen funcional será también grande y el término $(w^T x + b)$ grande y negativo. Además, si $y^{(i)}(w^T x + b) > 0$, entonces nuestra predicción en el ejemplo es correcta. Por lo tanto, un margen funcional grande representa una predicción fiable.

Para un clasificador lineal dado g (tomando valores en el intervalo $\{-1,1\}$), sin embargo se tiene una propiedad del margen funcional que no lo hace buen medidor de fiabilidad. Dado g , nótese que si se reemplaza w por $2w$ y b por $2b$, entonces se obtiene $g(w^T x + b) = g(2w^T x + 2b)$, que no haría cambiar $h_{w,b}(x)$ en absoluto. Tanto $h_{w,b}(x)$ como g , dependen solamente del signo no de la magnitud del término $(w^T x + b)$. Sin embargo, remplazar (w, b) por $(2w, 2b)$ también significa multiplicar nuestro margen funcional por un factor de 2. Así, parece que explotando el grado de libertad de la escala de (w, b) se puede hacer el margen funcional arbitrariamente grande sin cambiar nada significativamente. Intuitivamente, puede que tenga sentido imponer algún tipo de condición de normalización tal como $\|w\|_2 = 1$, sustituyendo (w, b) por $(w/\|w\|_2, b/\|w\|_2)$.

Dado un conjunto de ejemplo $S = \{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$, definiendo también el margen funcional de (w, b) con respecto a S como el menor de los márgenes funcionales de los conjuntos de ejemplos. Entonces se puede denotar $\hat{\gamma}$ como:

$$\hat{\gamma} = \min_{i=1, \dots, m} \hat{\gamma}^{(i)}.$$

Considere la figura siguiente:

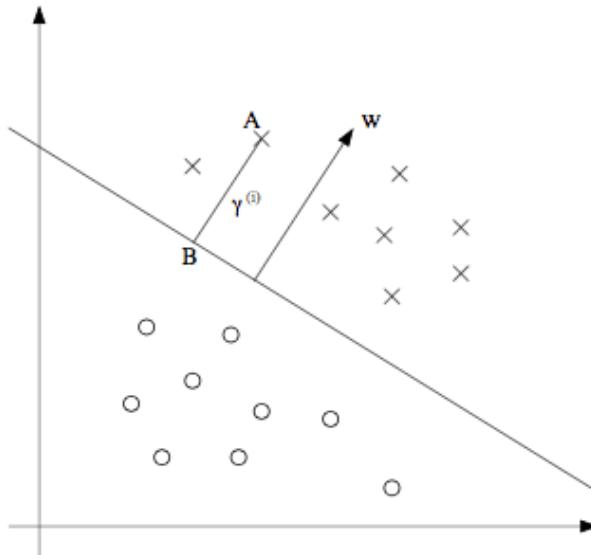


Figura 9 - Margen geométrico

El límite de decisión correspondiente a (w, b) se muestra como un plano ortogonal al vector w .

Considere el punto A, el cual representa la entrada $x^{(i)}$ de algunos ejemplos con la etiqueta $y^{(i)} = 1$. Su distancia al límite de decisión $\hat{\gamma}^{(i)}$ está dada por el segmento AB. Cómo se puede hallar el valor de $\hat{\gamma}^{(i)}$? ya que $w/\|w\|$ es un vector de longitud unidad apuntando a la misma dirección que w . Cómo A representa $x^{(i)}$, se tiene entonces que el punto B está dado por $x^{(i)} - \hat{\gamma}^{(i)} * w/\|w\|$. Pero este punto está situado justo en el límite de decisión, y todos los puntos x sobre dicho límite satisfacen la ecuación $(w^T x + b) = 0$. Por consiguiente,

$$w^T \left(x^{(i)} - \hat{\gamma}^{(i)} \frac{w}{\|w\|} \right) + b = 0$$

Resolviendo para $\hat{\gamma}^{(i)}$,

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{\|w\|} = \left(\frac{w}{\|w\|}\right)^T x^{(i)} + \frac{b}{\|w\|}$$

Este resultado sirve para casos de ejemplos positivos en A, donde el lado positivo es el lado "correcto". Más generalmente, se define el margen geométrico de (w, b) con respecto al ejemplo $(x^{(i)}, y^{(i)})$:

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \right)$$

Nótese que si $\|w\| = 1$, entonces el margen funcional es igual que al margen geométrico. Esto nos da una manera de relacionar estos dos tipos de márgenes. También, el margen geométrico es invariante reescalando los parámetros: si se usa $2w$ por w y $2b$ por b , entonces el margen geométrico no cambia. Debido a la invarianza de la escala de los parámetros, cuando se trata de hallar un w y b adecuados, se puede imponer una restricción de escala arbitraria en w sin cambiar nada importante. Por ejemplo, podemos forzar que $\|w\| = 1$, o que $|w_1 + b| + |w_2| = 2$, o cualquier otro puede satisfacerlo.

Finalmente, dado un conjunto de ejemplo $S = \{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$, definiendo también el margen funcional de (w, b) con respecto a S como el menor de los márgenes funcionales de los conjuntos de ejemplos. Entonces se puede denotar $\hat{\gamma}$ como:

$$\hat{\gamma} = \min_{i=1, \dots, m} \hat{\gamma}^{(i)}.$$

4.4 Margen clasificador óptimo

Dado un conjunto de prueba, nuestro objetivo obvio es encontrar un límite de decisión que maximice el margen, ya que esto reflejaría un conjunto de predicciones del conjunto de prueba más fiable. A grandes rasgos, esto sería un clasificador que separara los datos en positivos y negativos con un gran hueco entre ellos (margen geométrico).

El problema de maximización sería el siguiente:

$$\begin{aligned} & \max_{\gamma, w, b} \gamma \\ & \text{s. t. } y^{(i)} (w^T x^{(i)} + b) \geq \gamma, i = 1, \dots, m \\ & \|w\| = 1. \end{aligned}$$

La restricción última asegura que el margen funcional es igual al geométrico además de garantizar un margen geométrico de al menos γ . Por lo tanto, resolviendo este problema se asegura el margen geométrico mas amplio posible.

Lamentablemente, la última condición hace que sea inviable a nivel computacional, así que reformulamos el problema:

$$\begin{aligned} & \max_{\hat{\gamma}, w, b} (\hat{\gamma} / \|w\|) \\ & \text{s. t. } y^{(i)} (w^T x^{(i)} + b) \geq \hat{\gamma}, i = 1, \dots, m \end{aligned}$$

Aquí se maximizará $\hat{\gamma} / \|w\|$, sujeto al margen funcional siendo al menos $\hat{\gamma}$. Ya que el margen funcional y el geométrico están relacionados por $\gamma = \hat{\gamma} / \|w\|$, esto nos dará la respuesta que buscamos. Ahora, se introducirá una restricción de escala para (w, b) con respecto al conjunto de prueba que debe ser 1:

$$\hat{\gamma} = 1$$

Reformulando nuestro problema, obtenemos:

$$\min_{\gamma, w, b} \frac{1}{2} \|w\|^2$$

$$s.t. y^{(i)}(w^T x^{(i)} + b) \geq 1, i = 1, \dots, m$$

Así, se ha transformado el problema en una forma que puede ser resuelta eficientemente. La solución es el margen clasificador óptimo.

Sin embargo, conviene más en este tipo de optimización el uso de la dualidad de Lagrange que se verá a continuación.

4.5 Dualidad de Lagrange

Considere un problema de optimización de la forma:

$$\min_w f(w) \text{ s.t. } h_i(w) = 0, i = 0, \dots, l.$$

Que puede ser usado mediante multiplicadores de Lagrange. Se define el Lagrangiano como

$$\mathcal{L}(w, \beta) = f(w) + \sum_{i=1}^l \beta_i h_i(w)$$

Donde β_i son los multiplicadores de Lagrange. Para resolverlo trataríamos de hallar las derivadas parciales e igualarlas a cero.

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0; \frac{\partial \mathcal{L}}{\partial \beta_i} = 0,$$

y resolver para w y β .

Considere el siguiente problema de optimización al que llamaremos primal:

$$\begin{aligned} & \min_w f(w) \\ & \text{s.t. } g_i(w) \leq 0, i = 1, \dots, k \\ & \quad h_i(w) = 0, i = 1, \dots, l. \end{aligned}$$

Para resolverlo, empezamos definiendo el Lagrangiano generalizado:

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w)$$

Donde α_i e β_i son los multiplicadores de Lagrange. Considere también

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta)$$

Donde \mathcal{P} se refiere al primal. Si w viola cualquier restricción del primal, entonces se verificará que

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta: \alpha_i \geq 0} f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w) = \infty$$

Por el contrario, si las restricciones satisfacen la relación para un valor particular de w , entonces $\theta_{\mathcal{P}}(w) = f(w)$. Por lo tanto,

$$\theta_{\mathcal{P}}(w) = \begin{cases} f(w) & ; \text{si } w \text{ cumple las restricciones del primal} \\ \infty & ; \text{en otro caso} \end{cases}$$

Así, $\theta_{\mathcal{P}}$ toma el mismo valor que el objetivo en el problema para todos los valores de w que satisfacen las restricciones del primal, y es infinito positivo si la restricción no se cumple. Así, si se considera el problema de minimización

$$\min_w \theta_{\mathcal{P}}(w) = \min_w \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta)$$

Se ve que es el mismo problema que el inicial, el problema primal. Definimos el valor óptimo del objetivo como $p^* = \min_w \theta_{\mathcal{P}}(w)$

Ahora, se mirará otro problema ligeramente diferente. Se define

$$\theta_{\mathcal{D}}(w) = \min_w \mathcal{L}(w, \alpha, \beta)$$

Aquí, \mathcal{D} hace referencia a dual. Nótese también que como en la definición de $\theta_{\mathcal{P}}$ se estaba optimizando (maximizando) con respecto a α, β , ahora se está minimizando con respecto a w .

El problema dual de optimización será:

$$\max_{\alpha, \beta: \alpha_i \geq 0} \theta_{\mathcal{D}}(w) = \max_{\alpha, \beta: \alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta)$$

Este es exactamente el mismo que nuestro problema primal mostrado anteriormente, excepto el orden de los max y min están cambiados. También se define el valor óptimo del problema dual como $d^* = \max_{\alpha, \beta: \alpha_i \geq 0} \theta_{\mathcal{D}}(w)$.

Los problemas dual y primal están relacionados de la siguiente manera

$$d^* = \max_{\alpha, \beta: \alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta) \leq \min_w \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta) = p^*$$

Sin embargo, bajo ciertas condiciones tenemos que ambas soluciones son iguales $d^* = p^*$, así se podría resolver el problema dual ligado al primal. Suponga f y que los g_i son convexos, y que sus restricciones son factibles, esto significa que existe algún w que haga $g_i(w) < 0$ para todo i .

Bajo estas condiciones, existe w^*, α^*, β^* tales que w^* es la solución al problema primal, α^*, β^* son las soluciones del problema dual y además $p^* = d^* = \mathcal{L}(w^*, \alpha^*, \beta^*)$. También, tanto w^*, α^* como β^* satisfacen la condiciones de Karush-Kuhn-Tucker (KKT), que son como siguen:

$$\frac{\partial}{\partial w_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, i = 1, \dots, n \quad (1)$$

$$\frac{\partial}{\partial \beta_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, i = 1, \dots, l \quad (2)$$

$$\alpha_i^* g_i(w^*) = 0, i = 1, \dots, k \quad (3)$$

$$g_i(w^*) \leq 0, i = 1, \dots, k \quad (4)$$

$$\alpha^* \geq 0, i = 1, \dots, k \quad (5)$$

Especial atención merece la ecuación 5, que es llamada la condición complementaria dual. Implica que si $\alpha_i^* > 0$, entonces $g_i(w^*) = 0$. Que posee gran interés para estudios posteriores.

4.6 Clasificadores de márgenes óptimos

Previamente, definimos el problema primal de optimización para encontrar el clasificador de margen óptimo:

$$\begin{aligned} \min_{\gamma, w, b} \quad & \frac{1}{2} \|w\|^2 \\ \text{s.t. } & y^{(i)}(w^T x^{(i)} + b) \geq 1, i = 1, \dots, m \end{aligned}$$

Con las restricciones siguientes:

$$g_i(w) = -y^{(i)}(w^T x^{(i)} + b) + 1 \leq 0.$$

Una restricción por cada ejemplo de prueba. Nótese que desde la condición complementaria dual (ecuación (5)), se tendrá $\alpha_i > 0$ solo para aquellos ejemplos de prueba que tengan un margen funcional igual a uno. Considere la figura siguiente, con un hiperplano separador con margen máximo mostrado con una línea sólida.

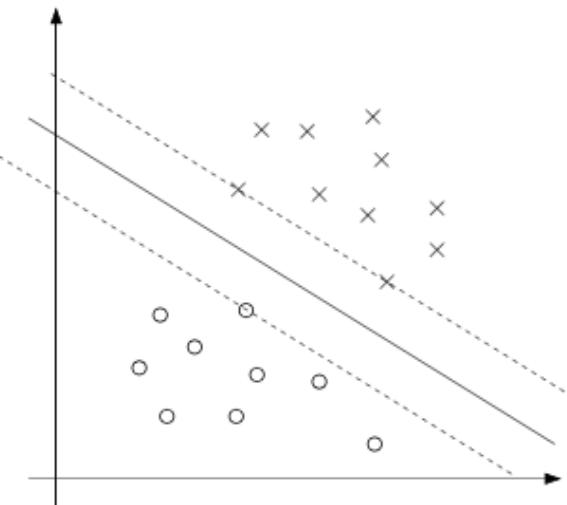


Figura 10 - hiperplano separador con margen máximo

Los puntos con los márgenes más pequeños son exactamente los más cercanos al límite de decisión, en nuestro caso son tres (uno negativo y dos positivos) que yacen sobre las líneas paralelas del límite de decisión.

Así, solo tres de los α_i serán distintos de cero en la solución óptima de nuestro problema. Estos tres vectores son llamados vectores de apoyo del problema. El hecho de que el número de vectores de apoyo pueda ser mucho menor que el tamaño del conjunto de prueba será bastante útil.

Se tratará ahora escribir nuestro algoritmo en términos solo del producto $\langle x^{(i)}, x^{(j)} \rangle$ entre puntos del espacio de entrada. Esto nos ayudará bastante en estudios posteriores.

Si construimos el Lagrangiano para nuestro problema de optimización se tiene:

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^m \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1] \quad (8)$$

Nótese que hay solo multiplicadores α_i y no β_i , ya que el problema tiene solo inecuaciones como restricciones.

Para encontrar el problema dual, se necesitará minimizar $\mathcal{L}(w, b, \alpha)$ con respecto a w y b (fijado α), para obtener θ_D , que se obtendrá mediante las derivadas parciales de \mathcal{L} con respecto a w y b igualando a cero. Se tendrá:

$$\nabla_w \mathcal{L}(w, b, \alpha) = w - \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} = 0$$

Esto implica que

$$w = \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} \quad (9)$$

Con la derivada con respecto a b , se obtiene

$$\frac{\partial}{\partial b} \mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i y^{(i)} = 0 \quad (10)$$

Si tomamos la definición de w de la ecuación 9, y la unimos con el Lagrangiano de la ecuación (8) y se simplifica se obtiene

$$\mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T x^{(j)} - b \sum_{i=1}^m \alpha_i y^{(i)}$$

Pero con la ecuación (10) el último término se hace cero, así pues

El problema dual de optimización se obtiene uniendo la ecuación anterior con la restricción de la

ecuación 10 y el hecho de que $\alpha_i \geq 0$, así se obtiene:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

$$\begin{aligned} s.t. \quad & \alpha_i \geq 0, i = 1, \dots, m \\ & \sum_{i=1}^m \alpha_i y^{(i)} = 0, \end{aligned}$$

Se debería poder verificar que las condiciones requeridas para que $p^* = d^*$ y las condiciones KKT se cumplen en el problema de optimización. Así pues, se podría calcular el problema dual ligado al primal. Específicamente, maximizando $W(\alpha)$ luego se podrá usar la ecuación (9) para encontrar el w óptimo como función de (α) . Habiendo encontrado w^* , considerando el problema primal, sería sencillo encontrar el valor óptimo para b como

$$b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + \min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2} \quad (11)$$

Antes de seguir, la ecuación (9) nos da el valor óptimo de w en términos de (α) . Suponga que se desea hacer una predicción en un nuevo punto de entrada x . Se calcularía $w^T x + b$, y se predice $y = 1$ si y solo si la cantidad es mayor que cero. Pero usando la ecuación (9), esta cantidad también puede ser escrita como:

$$w^T x + b = (\sum_{i=1}^m \alpha_i y^{(i)} x^{(i)})^T x + b \quad (12)$$

$$= \sum_{i=1}^m \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b \quad (13)$$

Por lo tanto, si encontramos los α , para hacer la predicción, se tendrá que calcular la cantidad que dependa solamente de del producto entre x y los puntos del conjunto de prueba. Ademas, como se comentó anteriormente, todos los α serán cero excepto los vectores de apoyo. Así, muchos términos de la suma serán cero, y solo se tendrá que calcular los términos del producto con los vectores de apoyo.

4.7 Kernels

Para el uso de kernels se cambiará la notación para cuando las variables de entrada se introduzcan en el algoritmo. Los conjuntos de variables de entrada son mapeados a nuevos conjuntos de variables, las cuales serán llamadas entradas características. Denominaremos también ϕ al mapeo característico, el cual realiza esta conversión. En nuestro ejemplo se tiene,

$$\phi(x) = \begin{bmatrix} x \\ x^2 \\ x^3 \end{bmatrix}$$

Antes que aplicar SVM usando los atributos de entrada x , se puede aprender ciertas carcterísticas de $\phi(x)$. Para ello, se necesita regresar al algoritmo previo y reemplazar x por $\phi(x)$.

Ya que el algoritmo puede ser escrito en términos de productos $\langle x, z \rangle$, esto significa que se reemplazará por $\langle \phi(x), \phi(z) \rangle$. Especificamente, dado el mapeo de ϕ , se define el kernel correspondiente

$$K(x, z) = \phi(x)^T \phi(z)$$

Entonces, donde antes se tenía previamente $\langle x, z \rangle$ en el algoritmo, se reemplazará simplemente por $K(x, z)$, y nuestro algoritmo estará preparado para las características de ϕ .

Ahora, dado ϕ , se puede calcular $K(x, z)$, hallando $\phi(x)$ y $\phi(z)$ y tomar el producto. Pero lo que es más interesante es que muy a menudo $K(x, z)$ puede ser muy factible de calcular, aun siendo $\phi(x)$

por si solo computacionalmente costoso, al ser un vector de alta dimensión. En tal caso, el uso de una forma eficiente de cálculo de $K(x, z)$ en el algoritmo, ayudará a entender el espacio multidimensional usando SVM, dado por ϕ , pero sin tener que encontrar explícitamente o representar vectores de $\phi(x)$.

Sea un ejemplo. Suponga $x, z \in \mathbb{R}^n$, y considere

$$K(x, z) = (x^T z)^2$$

También se puede escribir como

$$\begin{aligned} K(x, z) &= \left(\sum_{i=1}^n x_i z_i \right) \left(\sum_{j=1}^n x_j z_j \right) \\ &= \sum_{i=1}^n \sum_{j=1}^n x_i x_j z_i z_j \\ &= \sum_{i,j=1}^n (x_i x_j)(z_i z_j) \end{aligned}$$

Así, se ve que $K(x, z) = \phi(x)^T \phi(z)$, donde el mapeo ϕ está dado (para el caso de $n = 3$) por

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix}$$

Para un cierto Kernel, considere

$$\begin{aligned} K(x, z) &= (x^T z + c)^2 \\ &= \sum_{i,j=1}^n (x_i x_j)(z_i z_j) + \sum_{i=1}^n (\sqrt{2c} x_i)(\sqrt{2c} z_i) + c^2 \end{aligned}$$

Esto corresponde al mapeo característico para $n = 3$

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \\ \sqrt{2c} x_1 \\ \sqrt{2c} x_2 \\ \sqrt{2c} x_3 \\ c \end{bmatrix}$$

donde el parámetro c controla el coeficiente relativo entre los términos x_i (primer orden) y $x_i x_j$ (segundo orden).

Más ampliamente, el kernel $K(x, z) = (x^T z + c)^d$ corresponde al mapeo característico de un espacio $\binom{n+d}{n}$, correspondiente a todos los monomios del tipo $x_{i_1} x_{i_2} \dots x_{i_k}$ que son de hasta orden d . Sin

embargo, a pesar de funcionar en un espacio $O(n^d)$ -dimensional, calcular $K(x, z)$ todavía tarda $O(n)$ veces, y por lo tanto nunca se necesitará representar vectores en tales espacios dimensionales.

Intuitivamente, si $\phi(x)$ y $\phi(z)$ son muy parecidos entre sí, se podría esperar que $K(x, z) = \phi(x)^T \phi(z)$ fuese grande. Por el contrario, si $\phi(x)$ y $\phi(z)$ están muy separados (casi ortogonales entre sí), entonces $K(x, z) = \phi(x)^T \phi(z)$ será pequeño. Así, se podría pensar que $K(x, z)$ es como alguna medida de similaridad entre x y z .

Dicho esto, suponga que para un problema dado consigue un adecuado $K(x, z)$ para que x y z sean similares. Por ejemplo, una elección sería

$$K(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\sigma^2}\right)$$

Esta es una medida razonable para tener unos x y z parecidos, ya que se hace 1 cuando ambos son casi iguales, y cero cuando se alejan entre sí. Se podría aceptar esta definición de K como kernel en un SVM?^μ En este ejemplo particular, la respuesta sería sí. Este kernel es llamado Kernel Gaussiano, y corresponde a un mapeo característico de dimensión infinita de ϕ . Pero más ampliamente, dada una función K , ¿cómo se podría decir que es un kernel válido? ¿se podría decir que hay algún mapeo característico ϕ para que $K(x, z) = \phi(x)^T \phi(z)$ para todo x, z ?

Suponga ahora que K es de hecho un kernel válido correspondiente a algún mapeo característico ϕ . Considere ahora algún conjunto finito de m puntos $\{x^{(1)}, \dots, x^{(m)}\}$, y sea una matriz K cuadrada de orden m donde la entrada (i, j) -ésima está dada por $K_{ij} = K(x^{(i)}, x^{(j)})$. Esta matriz es llamada Matriz Kernel. Nótese que se ha llamado a la matriz con la misma notación que para la función kernel $K(x, z)$, debido a su obvia relación.

Ahora, si K es un kernel válido, entonces $K_{ij} = K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)}) = \phi(x^{(j)})^T \phi(x^{(i)}) = K(x^{(j)}, x^{(i)}) = K_{ji}$, y por tanto K debe ser simétrico. Además, sea $\phi_k(x)$ el k -ésimo vector coordinado $\phi(x)$, se encuentra entonces que para cualquier vector z ,

$$\begin{aligned} z^T K z &= \sum_i \sum_j z_i K_{ij} z_j \\ &= \sum_i \sum_j z_i \phi(x^{(i)})^T \phi(x^{(j)}) z_j \\ &= \sum_i \sum_j z_i \sum_k \phi_k(x^{(i)})^T \phi_k(x^{(j)}) z_j \\ &= \sum_i \sum_j \sum_k z_i \phi_k(x^{(i)})^T \phi_k(x^{(j)}) z_j \\ &= \sum_k (\sum_i z_i \phi_k(x^{(i)}))^2 \geq 0 \end{aligned}$$

Por lo tanto, se muestra que si K es un kernel válido, entonces le corresponde una matriz kernel $K \in \mathbb{R}^{m \times m}$ que es simétrica semidefinida positiva. Más generalmente, esto no es necesario pero sí suficiente para que K sea un kernel válido. Esto se afirma gracias al teorema de Mercer.

Queda visto la utilidad de esta idea de kernels para facilitar el estudio en SVM.

4.8 Regularización y el caso no separable

Hasta ahora se había asumido que la información era linealmente separable. Mientras que la

información mapeada a un espacio característico de altas dimensiones por ϕ generalmente incrementa la probabilidad de la separabilidad de los datos, no se garantiza que siempre sea así. También, en algunos casos no está claro si encontrar un hiperplano separador es exactamente lo que se necesita ya que sería susceptible a valores extremos. Por ejemplo, la figura de abajo a la izquierda muestra un clasificador de márgenes óptimo, y cuando se añade un solo valor extremo en la región superior izquierda (figura derecha), causa que el límite de decisión haga un oscilación significativo. El resultado es un clasificador con un margen mucho más pequeño.

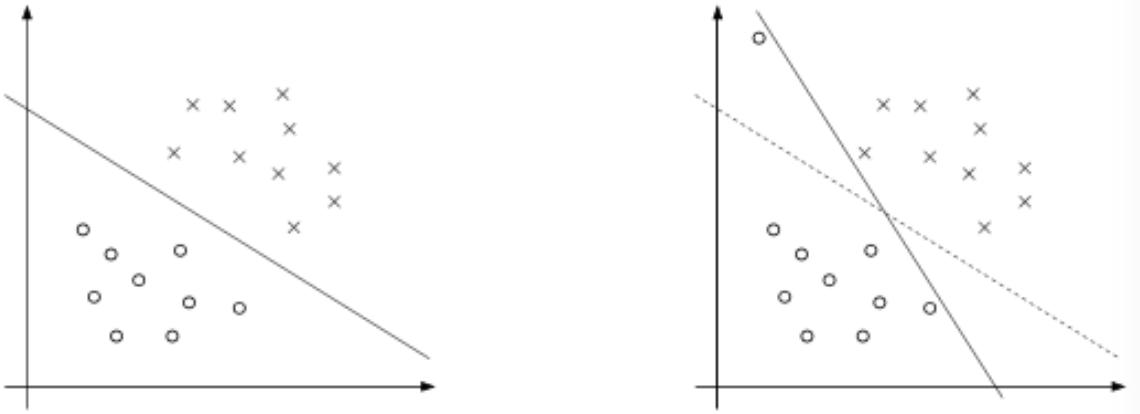


Figura 11 – Regularización y caso no separable

Para que el algoritmo funcione en situaciones de conjuntos de datos no lineales y no separables, reformulamos el problema de optimización:

$$\begin{aligned} \min_{w,b} & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i \\ \text{s.t. } & y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, i = 1, \dots, m \\ & \xi_i \geq 0, i = 1, \dots, m \end{aligned}$$

Así, las muestras tendrán un margen funcional menor a uno, y si una muestra tiene un margen funcional de $1 - \xi_i$ con $\xi_i > 0$, se vería como la función objetivo del problema se incrementa en $C\xi_i$. El parámetro C controla el coeficiente entre los objetivos de minimizar $\|w\|^2$ y asegurar que la mayoría de las muestras tienen un margen funcional de al menos 1.

Como antes, reformulamos el Lagrangiano:

$$L(w, b, \xi, \alpha, r) = \frac{1}{2} w^T w + C \sum_{i=1}^m \xi_i - \sum_{i=1}^m \alpha_i [y^{(i)}(x^T w + b) - 1 + \xi_i] - \sum_{i=1}^m r_i \xi_i$$

Donde los multiplicadores de Lagrange (limitados a positivos) son α_i y r_i . Despues de derivar el dual y todos los pasos posteriores y simplificar, se obtiene la forma dual del problema:

$$\begin{aligned} \max_{\alpha} W(\alpha) &= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \\ \text{s.t. } & 0 \leq \alpha_i \leq C, i = 1, \dots, m \\ & \sum_{i=1}^m \alpha_i y^{(i)} = 0, \end{aligned}$$

Como antes, también se tiene que expresar w en términos de α_i , como se vio anteriormente. Así, despues de solucionar el problema dual se nota que el único cambio con lo anterior es el factor de la regularización pero solamente en la restriccion que era originalmente $0 \leq \alpha_i$ ahora se convierte en $0 \leq \alpha_i \leq C$. Además, las condiciones de complementariedad KKT se ven afectadas como sigue:

$$\alpha_i = 0 \Rightarrow y^{(i)}(w^T x^{(i)} + b) \geq 1$$

$$\alpha_i = C \Rightarrow y^{(i)}(w^T x^{(i)} + b) \leq 1$$

$$0 < \alpha_i < C \Rightarrow y^{(i)}(w^T x^{(i)} + b) = 1$$

Lo restante es dar con el algoritmo que realmente resuelva el problema dual que se verá en el siguiente apartado.

4.9 Algoritmo SMO

El algoritmo SMO (optimización secuencial mínima), nos da una manera eficiente de resolver el problema dual que surge del SVM

4.9.1 Coordenadas

Considere el problema sin restricciones de optimización

$$\max_{\alpha} W(\alpha_1, \alpha_2, \dots, \alpha_m)$$

Aquí, se considerará que W es solo una función de parámetros α_i y que no tiene relación con el problema de SVM. El algoritmo de optimización que se considerará aquí es el llamado “subida coordinada”:

Bucle hasta que converga {

For i = 1, ..., m, {

$$\alpha_i := \arg \max_{\hat{\alpha}_i} W(\alpha_1, \dots, \alpha_{i-1}, \hat{\alpha}_i, \alpha_{i+1}, \dots, \alpha_m)$$

}

}

Dentro del bucle de este algoritmo, se mantendrán todas las variables excepto algunos α_i fijos y se reoptimiza para un solo parámetro α_i . En esta versión del algoritmo, se presenta el bucle reoptimizando las variables en orden $\alpha_1, \alpha_2, \dots, \alpha_m, \alpha_1, \alpha_2, \dots$ las versiones más avanzadas del algoritmo usan otro tipo de ordenamiento.

Cuando la función W resulta ser de tal forma que el argumento máximo dentro del bucle puede ser ejecutado eficientemente, entonces la subida coordinada puede ser un algoritmo bastante eficiente.

En la siguiente figura se muestra gráficamente el procedimiento del algoritmo.

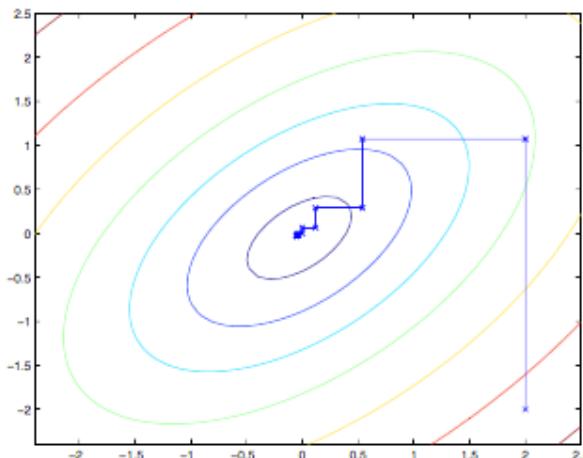


Figura 12 - Algoritmo de subida coordinada

La elipse de la figura son los contornos de una función cuadrática la cual se desea optimizar. La

subida coordinada se inicializó en (2,-2), y también se muestra el camino que toma para llegar al máximo global. Nótese que en cada paso, toma un paso que es paralelo a uno de los ejes, ya que solo una variable está siendo optimizada al mismo tiempo.

4.9.2 SMO

Este es el problema dual de optimización que se quiere resolver:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

$$s.t. 0 \leq \alpha_i \leq C, i = 1, \dots, m$$

$$\sum_{i=1}^m \alpha_i y^{(i)} = 0.$$

Se tiene un conjunto de α_i que satisface las dos últimas restricciones. Ahora, supóngase que se quiere mantener fijas $\alpha_2, \dots, \alpha_m$, y dar un paso de ascenso coordinado y volver a optimizar el objetivo con respecto a α_1 . Pero no supondría un gran cambio porque la restricción última asegura que

$$\alpha_1 y^{(1)} = - \sum_{i=2}^m \alpha_i y^{(i)}$$

O, también multiplicando ambos lados por $y^{(1)}$, se tiene

$$\alpha_1 = -y^{(1)} \sum_{i=2}^m \alpha_i y^{(i)}$$

Por lo tanto, α_1 está exactamente determinado por los otros α_i , y si hubiésemos dejado fijos $\alpha_2, \dots, \alpha_m$, entonces no podríamos hacer ningún cambio a α_1 sin violar la restricción última del problema de optimización.

Por lo tanto, si quisieramos actualizar los α_i , deberíamos actualizar al menos dos de ellos simultáneamente para así satisfacer las restricciones. El algoritmo SMO hace lo siguiente:

Repetir hasta converger {

1. Seleccionar un par de α_i y de α_j para actualizar (usando una heurística que trata de escoger los dos que nos permitirán hacer el progreso mayor hacia el máximo global).
2. Reoptimiza $W(\alpha)$ con respecto a α_i y α_j , mientras todas las demás α_k ($k \neq i, j$) son fijas.

}

Para probar la convergencia de este algoritmo, se puede ver si las condiciones KKT se satisfacen dentro de un tol . Un tol es el parámetro de tolerancia de convergencia, y típicamente se establece cerca de 0.01 hasta 0.001.

La razón más importante que hace de SMO un algoritmo eficiente es que la actualización de α_i y de α_j pueda ser calculada de forma efectiva.

Actualmente, se tiene un conjunto de configuraciones de α_i que satisfacen las restricciones, y supóngase que se decide mantener fijos $\alpha_3, \dots, \alpha_m$, y se quiere reoptimizar $W(\alpha_1, \alpha_2, \dots, \alpha_m)$ con respecto a α_1 y α_2 (sujeto a las restricciones). De esto último, se requiere que

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = - \sum_{i=3}^m \alpha_i y^{(i)}$$

Ya que el miembro de la derecha es fijo, se puede definir como una constante, llamémosla ζ .

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = \zeta$$

Se puede representar las restricciones en α_1 y α_2 como sigue:

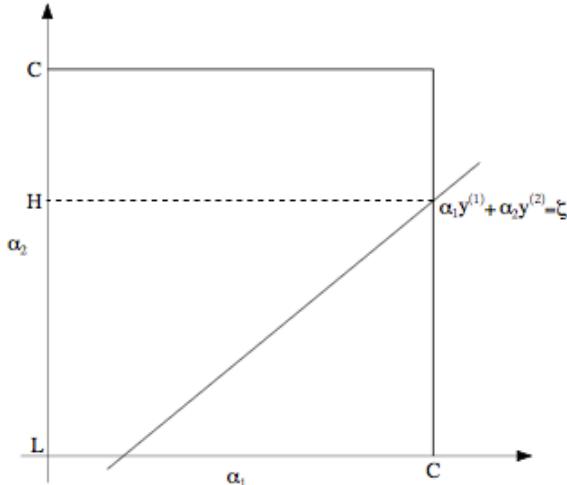


Figura 13 - Restricciones algoritmo SMO

De las restricciones del problema dual se sabe que α_1 y α_2 están en el rango $[0, C] \times [0, C]$, como se muestra en la imagen. También se ha mostrado en la imagen la línea $\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = \zeta$. Nótese también que de estas restricciones se sabe que $L \leq \alpha_2 \leq H$, de lo contrario (α_1, α_2) no podría satisfacer simultáneamente la recta y el rango de la caja. En este ejemplo, $L = 0$. Pero dependiendo de cómo sea la línea $\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = \zeta$, no será siempre el caso. Mas generalmente, habrá algún límite inferior de L y algún límite superior de H dentro de los posibles valores de α_2 que aseguren que α_1, α_2 están en el rango de la caja.

Reformulando la ecuación anterior, como función de α_2

$$\alpha_1 = (\zeta - \alpha_2 y^{(2)}) y^{(1)}$$

Por lo tanto, el objetivo $W(\alpha)$ puede ser escrito como

$$W(\alpha_1, \alpha_2, \dots, \alpha_m) = W((\zeta - \alpha_2 y^{(2)}) y^{(1)}, \alpha_2, \dots, \alpha_m)$$

Tratando a $\alpha_3, \dots, \alpha_m$ como constantes, se debería ser capaz de verificar que es una función cuadrática en α_2 . Esto puede ser expresado de la forma básica de una ecuación de segundo grado. Si ignoramos las restricciones que implican la caja, se podría fácilmente maximizar la función cuadrática haciendo su derivada cero y resolver.

Sea $\alpha_2^{unclipped}$ el mencionado valor de α_2 . Se debería ser capaz de ver que si se hubiese querido maximizar W con respecto a α_2 pero sujeto a las restricciones de la caja, entonces resultaría que el valor óptimo simplemente tomando $\alpha_2^{unclipped}$ y "recortarlo" para que entre en el intervalo $[L, H]$ para así,

$$\alpha_2^{new} = \begin{cases} H & , si \alpha_2^{unclipped} > H \\ \alpha_2^{unclipped} & , si L \leq \alpha_2^{unclipped} \leq H \\ L & , si \alpha_2^{unclipped} < L \end{cases}$$

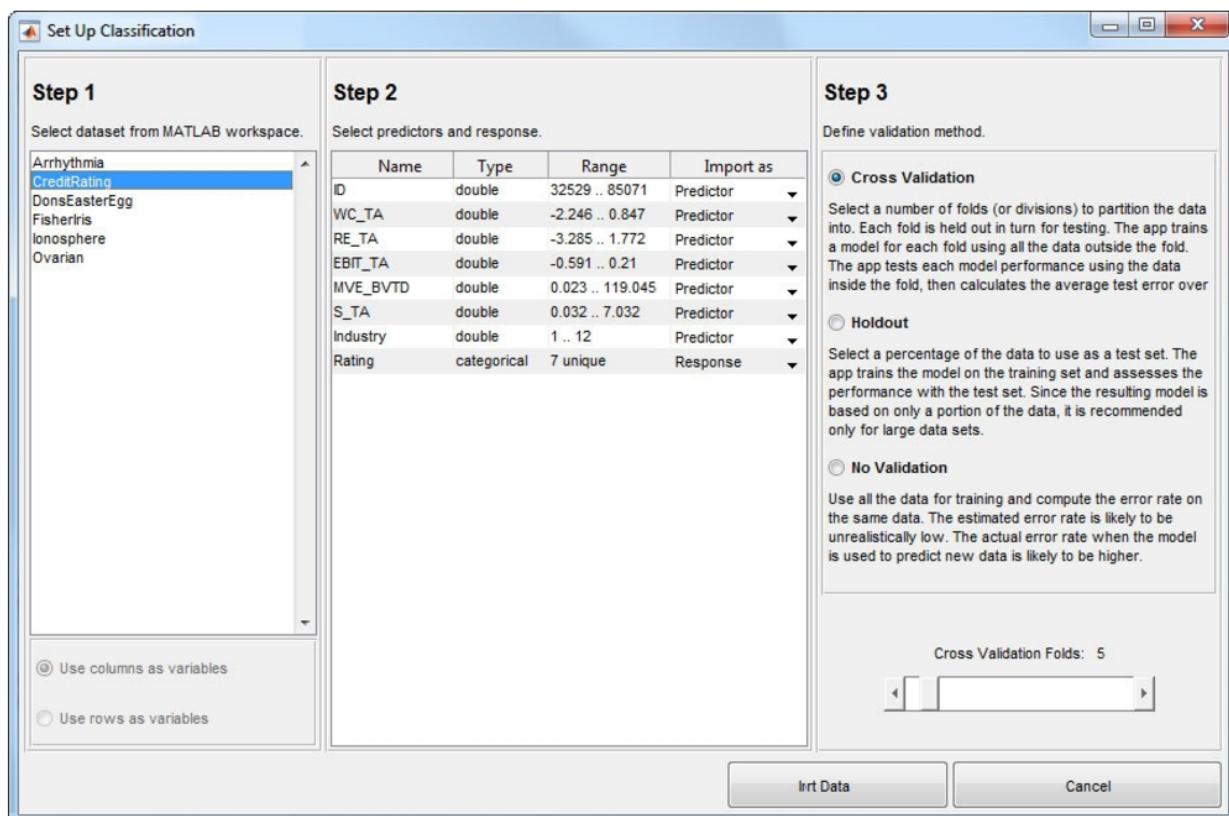
Finalmente, habiendo encontrado α_2^{new} , se puede usar la ecuación de ζ para volver y encontrar el

valor óptimo de α_1^{new} .

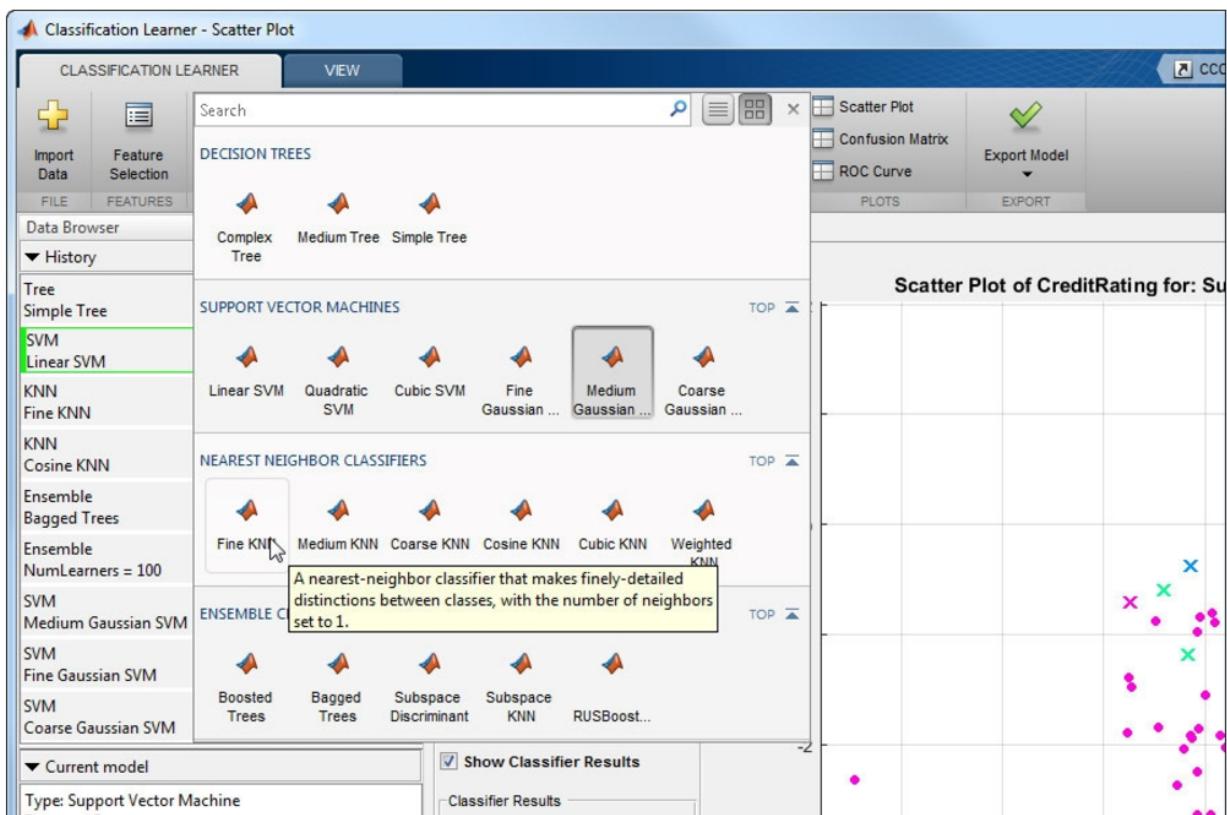
4.10 Aplicación Matlab para MSV

Para la realización de las máquinas de soporte vectorial he escogido la aplicación que incluye Matlab llamado 'Classification Learner', el cual entrena modelos para clasificar datos. Utilizando esta aplicación, se puede explorar sobre el aprendizaje maquina utilizando varios clasificadores. Se pueden explorar datos, seleccionar características, especificar esquemas de validación, formar modelos y evaluar los resultados. Se puede realizar un entrenamiento automatizado para buscar el mejor tipo de modelo de clasificación, incluyendo árboles de decisión, análisis discriminante, máquinas de soporte vectorial, regresión logística, vecinos más cercanos y clasificación de conjuntos. Se puede realizar un aprendizaje maquina suministrando un conjunto conocido de datos de entrada (observaciones o ejemplos) y respuestas conocidas a los datos (por ejemplo, etiquetas o clases). Los datos se utilizan para formar un modelo que genera predicciones para la respuesta a nuevos datos. Para utilizar el modelo con nuevos datos, o para aprender sobre la clasificación programada, puede exportar el modelo al área de trabajo o generar un código Matlab para recrear el modelo entrenado.

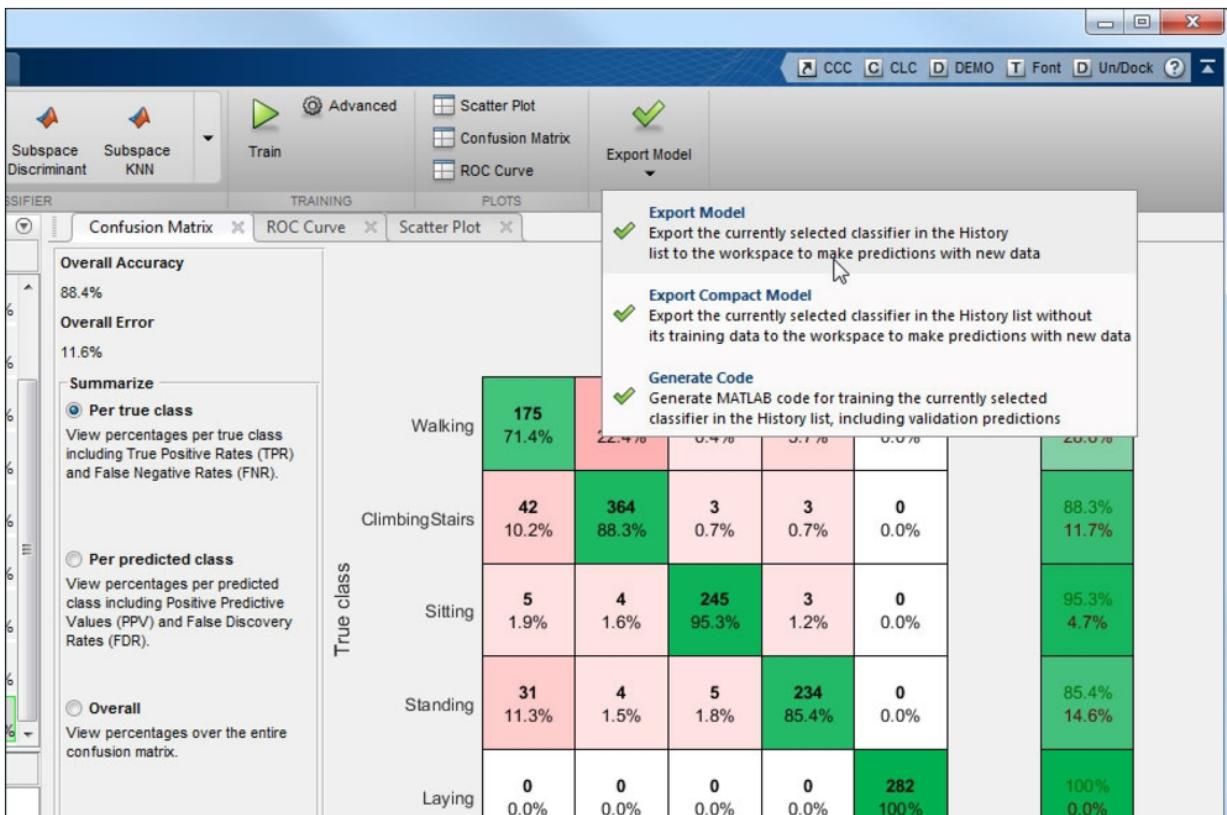
El primer paso es elegir el vector o la matriz que servirá para entrenar a la MSV, cada variable puede ser elegida como respuesta o predictor. Predictor son aquellos vectores que se clasificarán mientras que respuesta es aquél vector que sirve para indicar una característica que es la que clasifica a los predictores, esto se puede ver en la siguiente figura.

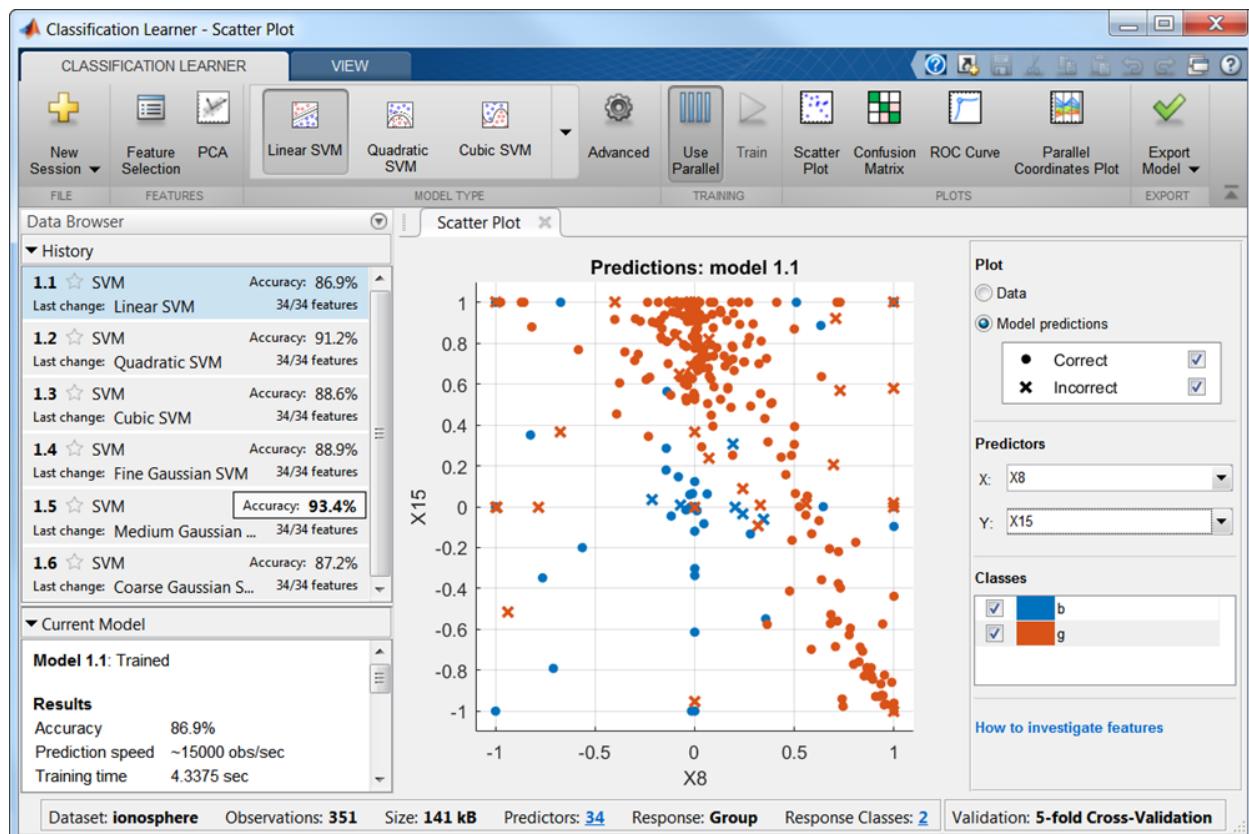


Después, se elige el tipo de clasificador que se desea y se entrena, como se ve en la figura siguiente. Se pueden entrenar con distintos clasificadores y luego elegir el que tiene el desempeño mayor.



Existen también otras utilidades interesantes como la matriz de confusión de los datos o la posibilidad de exportar la MSV al directorio como archivo independiente o como variables directamente. Con esto ultimo se podrá hacer nuevas predicciones con otros datos.





5 PRUEBAS

“Para saber hablar es preciso saber escuchar.”

- Plutarco, Historiador griego -

En este capítulo se presentará las pruebas realizadas, además en el anexo se encuentran los script escritos con el programa de computación Matlab.

Se han realizado nueve máquinas de soporte vectorial distintas: tres entrenadas con datos provenientes del coeficiente de asimetría para cada banda de frecuencia, otras tres entrenadas con datos provenientes de la curtosis en cada banda y finalmente, tres entrenadas con datos provenientes de la potencia para cada banda pero con información limitada. Los vectores con la información nula corresponden a las bandas 100, 150, 200 y 300.

Los datos usados para la creación de las máquinas de soporte virtual tanto del coeficiente de asimetría como para el de curtosis son relativos del paciente 1 en su registro 2, mientras que la máquina de soporte vectorial de potencia limitada son relativos al paciente 3 en su registro 3.

Para cada paciente se han estudiado cinco registros del encefalograma, cada uno de una hora de duración menos el paciente 6 que duran cuatro horas.

La duración de todos los bloques es de 10 segundos en todos los registros.

Así, en cada registro podremos tener una crisis, ninguna crisis o una crisis temprana, dependiendo del momento en el que sucede la crisis dentro del registro. Esto será importante para los resultados ya que dependiendo de cuanto antes empecemos a estudiar el registro tendremos más o menos probabilidades de prever la crisis.

Los resultados de estas pruebas serán comentados en el siguiente capítulo.

5.1 Paciente 1

5.1.1 Registro 1

El registro 1 del paciente uno no tiene ninguna crisis epiléptica.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.1.2 Registro 2

El paciente 1 tiene una crisis epiléptica a los 2996 segundos que corresponde al bloque 300 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 100 para conseguir mayor precisión, esto significa empezar en el segundo 996 antes del ataque epiléptico.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenando con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 55.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 57.222222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 54.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.1.3 Registro 3

El paciente 1 tiene una crisis epiléptica a los 1467 segundos que corresponde al bloque 147 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 7 para maximizar la precisión, esto significa empezar en el segundo 67 antes del ataque epiléptico hasta el inicio de la crisis. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 99.444444 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 50.555556 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda

alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Misma probabilidad de crisis que no crisis por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 50.277778 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.1.4 Registro 4

El registro 4 del paciente 1 no tiene ninguna crisis epiléptica.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.1.5 Registro 5

El paciente 1 tiene una crisis epiléptica a los 1732 segundos que corresponde al bloque 174 de los 360 que en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 24 para maximizar la precisión, esto significa empezar en el segundo 232 antes del ataque epiléptico. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 92.222222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 62.222222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 61.944444 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 62.222222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.2 Paciente 2

5.2.1 Registro 1

El registro 1 del paciente 2 no tiene ninguna crisis epiléptica. Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.2.2 Registro 2

El paciente 2 tiene una crisis epiléptica a los 130 segundos que corresponde al bloque 13 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 3 para maximizar la precisión, esto significa empezar en el segundo 30 antes del ataque epiléptico hasta el inicio de la crisis. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetria y entrenándolo con Coeficiente de asimetria en banda alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetria y entrenándolo con Coeficiente de asimetria en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetria y entrenándolo con Coeficiente de asimetria en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma

5.2.3 Registro 3

El paciente 2 tiene una crisis epiléptica a los 2998 segundos que corresponde al bloque 300 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 100 para conseguir mayor precisión, esto significa empezar en el segundo 998 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 71.944444 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 60.833333 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 58.888889 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 58.055556 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 55.833333 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma

5.2.4 Registro 4

El paciente 2 tiene una crisis epiléptica a los 3396 segundos que corresponde al bloque 337 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 137 para maximizar la precisión, esto significa empezar en el segundo 1396 antes del ataque epiléptico.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 50.555556 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 52.500000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 52.500000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 50.833333 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.2.5 Registro 5

El registro 3 del paciente dos no tiene ninguna crisis epiléptica.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma

- Potencia alfa

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma

- Potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.3 Paciente 3

5.3.1 Registro 1

El paciente 3 tiene una crisis epiléptica a los 362 segundos que corresponde al bloque 37 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 7 para maximizar la precisión, esto significa empezar en el segundo 62 antes del ataque epiléptico hasta el inicio de la crisis. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 65.555556 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 60.555556 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 82.222222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.3.2 Registro 2

El paciente 3 tiene una crisis epiléptica a los 721 segundos que corresponde al bloque 74 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 4 para maximizar la precisión, esto significa empezar en el segundo 21 antes del ataque epiléptico hasta el inicio de la crisis. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 85.555556 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 99.444444 por ciento usando como SVM Coeficiente de

asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 50.833333 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 51.388889 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 52.222222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.3.3 Registro 3

El paciente 3 tiene una crisis epiléptica a los 2162 segundos que corresponde al bloque 217 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 17 para maximizar la precisión, esto significa empezar en el segundo 162 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 57.777778 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.3.4 Registro 4

El registro 3 del paciente tres no tiene ninguna crisis epiléptica. Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.3.5 Registro 5

El paciente 3 tiene una crisis epiléptica a los 2592 segundos que corresponde al bloque 260 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 60 para maximizar la precisión, esto significa empezar en el segundo 592 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 56.944444 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 60.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 56.944444 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa

No hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma

5.4 Paciente 4

5.4.1 Registro 1

El registro 1 del paciente 4 no tiene ninguna crisis epiléptica. Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma

5.4.2 Registro 2

El paciente 4 tiene una crisis epiléptica a los 417 segundos que corresponde al bloque 42 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 2 para maximizar la precisión, esto significa empezar en el segundo 17 antes del ataque epiléptico hasta el inicio de la crisis. Como se nota, es casi al principio del registro.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 64.444444 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 67.222222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 66.944444 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.4.3 Registro 3

El paciente 4 tiene una crisis epiléptica a los 2451 segundos que corresponde al bloque 246 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el bloque 46 para maximizar la precisión, esto significa empezar en el segundo 451 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma..

- Coeficiente de asimetría

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 59.166667 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 50.277778 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa%

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta%

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 55.833333 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.4.4 Registro 4

El paciente 4 tiene una crisis epiléptica a los 2348 segundos que corresponde al bloque 235 de los 360 que hay en este registro de una hora. Se ha propuesto que el estudio de la predicción comience en el

bloque 35 para maximizar la precisión, esto significa empezar en el segundo 348 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 93.333333 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis beta..

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 57.777778 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.444444 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 59.444444 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.166667 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 59.166667 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.4.5 Registro 5

El registro 5 del paciente 4 no tiene ninguna crisis epiléptica. Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y

entrenándolo con curtosis beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 99.722222 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 98.888889 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

No hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.5 Paciente 5

5.5.1 Registro 1

El paciente 4 tiene una crisis epiléptica a los 12500 segundos que corresponde al bloque 1084 de los 1440 que hay en este registro de cuatro hora. Se ha propuesto que el estudio de la predicción comience en el bloque 284 para maximizar la precisión, esto significa empezar en el segundo 4500 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta.

Hay crisis con una probabilidad de 99.930507 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma.

- Potencia alfa

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.930507 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.930507 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

5.5.2 Registro 2

El paciente 4 tiene una crisis epiléptica a los 10833 segundos que corresponde al bloque 1250 de los 1440 que hay en este registro de cuatro horas. Se ha propuesto que el estudio de la predicción comience en el bloque 450 para maximizar la precisión, esto significa empezar en el segundo 2833 antes del ataque epiléptico hasta el inicio de la crisis.

Si se procesa este registro se obtiene, dependiendo de la MSV que sea, lo siguiente:

- Curtosis

No hay crisis con una probabilidad de 54.343294 por ciento usando como SVM curtosis y entrenándolo con curtosis alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM curtosis y entrenándolo con curtosis beta.

Hay crisis con una probabilidad de 99.861015 por ciento usando como SVM curtosis y entrenándolo con curtosis gamma.

- Coeficiente de asimetría

Hay crisis con una probabilidad de 63.585823 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda alfa

Hay crisis con una probabilidad de 99.861015 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda beta

Hay crisis con una probabilidad de 99.861015 por ciento usando como SVM Coeficiente de asimetría y entrenándolo con Coeficiente de asimetría en banda gamma

- Potencia alfa

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.861015 por ciento usando como SVM potencia en banda alfa con información limitada y entrenándolo con potencia gamma.

- Potencia beta

Hay crisis con una probabilidad de 51.633079 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.930507 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.791522 por ciento usando como SVM potencia en banda beta con información limitada y entrenándolo con potencia gamma.

- Potencia gamma

Hay crisis con una probabilidad de 100.000000 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia alfa.

Hay crisis con una probabilidad de 99.930507 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia beta.

Hay crisis con una probabilidad de 99.861015 por ciento usando como SVM potencia en banda gamma con información limitada y entrenándolo con potencia gamma.

6 RESULTADOS

"La primera virtud es frenar la lengua, y es casi un dios quien teniendo razón sabe callarse."

- Marco Porcio Catón 'el Joven', Político de la república Romana-

Una vez mostrados los resultados de las simulaciones en el capítulo anterior, se comentarán tales resultados para sacar conclusiones.

Para empezar, se ha visto que cuanto más temprana es la crisis epiléptica dentro del registro, más complicado será prever el ataque ya que si es así las máquinas de soporte vectoriales no son capaces de distinguir entre crisis o no crisis, al no tener suficiente información previa. Este hecho resulta sumamente importante, ya que hace de la elección de los segundos previos al ataque un parámetro crucial.

Esto se puede observar por ejemplo en el registro 2 del paciente cuatro o en los registros 1 y 2 del paciente tres. Todos ellos con una crisis antes del segundo 1000 del registro. En este tipo de registros se ve que la banda de uso mejor es la banda alfa aunque no queda claro que SVM usar, ya que si en el paciente dos la SVM más fiable sería la potencia en banda alfa (que ni siquiera acierta al final pero es el que más se acerca), en el caso del paciente tres aciertan tanto la potencia como la curtosis y el coeficiente de asimetría, todos en banda alfa. A medida de que la crisis es más avanzada en el tiempo, como en el registro 2 del paciente tres, se va cambiando esta tendencia hacia las SVM gamma y beta siempre entrenadas con banda alfa.

Un ejemplo más claro de la importancia de la elección correcta de los segundos previos de estudio son las crisis en los registros con en torno a los 1700 segundos. Por ejemplo los registros 3 y 5 del paciente uno donde las crisis aparecen en torno a 1400 segundos y 1700 segundos respectivamente. En estos casos se puede observar que la SVM más fiable es la curtosis en banda alfa en ambos casos. Sin embargo, para el registro 5 con una crisis más tardía, son más eficaces las SVM de potencia entrenadas por la potencia en banda gamma.

Para las crisis que acaecen más allá de los 2000 segundos se puede afirmar con seguridad que las mejores SVM son las entrenadas con banda gamma y banda beta, siendo la alfa desaconsejable.

En general, para prever una crisis epiléptica con unos registros de estas características se debería usar entre 1600 y 2200 segundos como tiempo de estudio previo al ataque. Esto obviamente para registros de una hora. Para el paciente cinco con registros de cuatro horas este tiempo se multiplica por cuatro.

Generalizando, como se decía anteriormente el mejor filtro por bandas a utilizar para la detección anticipada de crisis epilépticas es el filtro Gamma. Éste utiliza el rango de frecuencias mayores a 30 Hz. Le sigue el filtro beta y por último Alfa que es el peor filtro para la detección en condiciones normales, pero como ya se comentó funciona razonablemente bien para crisis tempranas.

Otro aspecto a tener en cuenta es la necesidad de estudiar cada paciente por separado, ya que las

crisis surgen en zonas diferentes del cerebro y luego se propagan. Esto influye en la creación de las SVM, como se han creado a partir de la información de un paciente, esta SVM puede no ser válida (más o menos) para otro paciente.

Como líneas futuras de trabajo se pueden añadir nuevas bandas de frecuencias como la delta o la theta y ver si son bandas más aptas para la prevención. También sería posible estudiar si mejoran los resultados con otras SVM, como por ejemplo una entrenada con una crisis temprana, otra que tenga menos vectores con información u otra que contenga todos los datos a la vez.

8 REFERENCIAS

- [1] Stephen A. Zahorian and Hongbing Hu,
“Nonlinear Dimensionality Reduction Methods for Use with Automatic Speech Recognition”
- [2] López Pisón J, Dolz Zaera I, Arana Navarro T. “*El electroencefalograma en el estudio y control de la epilepsia*”. Form Act Pediatr Aten Prim.2013;6:216-26
- [3] Ali Hossam Shoeb, “*Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment*”
- [4] Ali Shoeb, Herman Edwards, Jack Connolly, Blaise Bourgeois, S. Ted Treves and John Guttag, “*Patient-specific seizure onset detection*”
- [5] Andrew Ng, “*CS229 Lecture notes*”
- [6] Mathworks Matlab. Computer Program. Mathworks, 2016
- [7] Lara V. Marcuse, Madeline C. Fields, Jiyeoun (Jenna) Yoo, *Rowan's primer of EEG*

7 ANEXOS

Se presenta a continuación los scripts en lenguaje Matlab realizados para el desarrollo del proyecto.

7.1 Script 1: *bloques.m*

```
%  
% Andres Garcia-Baquero Leon  
%  
% TFM: Procesamiento de senal de un EEG para pacientes epilepticos  
%  
  
clear  
close all  
tic  
  
disp('Pacientes disponibles: 1, 2, 3, 5 y 6')  
opcion = input('Ingrese el numero del paciente que desea cargar: ');  
switch opcion  
case 1  
disp('Informacion del paciente 1 disponibles: 1, 3, 4, 10 y 15')  
opcion2 = input('Ingrese la informacion del paciente 1 que desea cargar: ');  
switch opcion2  
case 1  
load('chb01_01_edfm.mat') %cargamos la informacion del eeg  
case 3  
load('chb01_03_edfm.mat') %cargamos la informacion del eeg  
case 4  
load('chb01_04_edfm.mat') %cargamos la informacion del eeg  
case 10  
load('chb01_10_edfm.mat') %cargamos la informacion del eeg  
case 15  
load('chb01_15_edfm.mat') %cargamos la informacion del eeg  
otherwise  
disp('dato introducido erroneo')  
end  
case 2  
disp('Informacion del paciente 2 disponibles: 10, 16, 17, 19 y 35')  
opcion2 = input('Ingrese la informacion del paciente 2 que desea cargar: ');  
switch opcion2  
case 10  
load('chb02_10_edfm.mat') %cargamos la informacion del eeg  
case 16  
load('chb02_16_edfm.mat') %cargamos la informacion del eeg  
case 17  
load('chb02_17_edfm.mat') %cargamos la informacion del eeg  
case 19  
load('chb02_19_edfm.mat') %cargamos la informacion del eeg  
case 35  
load('chb02_35_edfm.mat') %cargamos la informacion del eeg  
otherwise  
disp('dato introducido erroneo')  
end  
case 3  
disp('Informacion del paciente 3 disponibles: 1, 2, 4, 5 y 35')  
opcion2 = input('Ingrese la informacion del paciente 3 que desea cargar: ');  
switch opcion2
```

```

case 1
load('chb03_01_edfm.mat') %cargamos la informacion del eeg
case 2
load('chb03_02_edfm.mat') %cargamos la informacion del eeg
case 4
load('chb03_04_edfm.mat') %cargamos la informacion del eeg
case 5
load('chb03_5_edfm.mat') %cargamos la informacion del eeg
case 35
load('chb03_35_edfm.mat') %cargamos la informacion del eeg
otherwise
disp('dato introducido erroneo')
end
case 5
disp('Informacion del paciente 5 disponibles: 1, 6, 17, 22 y 39')
opcion2 = input('Ingrese la informacion del paciente 5 que desea cargar: ');
switch opcion2
case 1
load('chb05_01_edfm.mat') %cargamos la informacion del eeg
case 6
load('chb05_06_edfm.mat') %cargamos la informacion del eeg
case 17
load('chb05_17_edfm.mat') %cargamos la informacion del eeg
case 22
load('chb05_22_edfm.mat') %cargamos la informacion del eeg
case 39
load('chb05_39_edfm.mat') %cargamos la informacion del eeg
otherwise
disp('dato introducido erroneo')
end
case 6
disp('Informacion del paciente 6 disponibles: 9 y 10')
opcion2 = input('Ingrese la informacion del paciente 6 que desea cargar: ');
switch opcion2
case 9
load('chb06_09_edfm.mat') %cargamos la informacion del eeg
case 10
load('chb06_10_edfm.mat') %cargamos la informacion del eeg
otherwise
disp('dato introducido erroneo')
end
otherwise
disp('dato introducido erroneo')
end

if(opcion==6)
    time=14400;%4horas
else
time=3600; %en seg (1h)
end

time_crisis1_1=3600;%segundos
time_crisis1_3=2996;%segundos
time_crisis1_4=1467;%segundos
time_crisis1_15=1732;%segundos
time_crisis2_19=3369;%segundos
time_crisis5_22=2348;%segundos
time_crisis2_16=130;%segundos
time_crisis2_17=2988;%segundos
time_crisis3_1=362;
time_crisis3_4=2162;
time_crisis3_35=2592;
time_crisis3_2=731;
time_crisis5_6=417;

```

```

time_crisis5_17=2451;
time_crisis6_9=12500;
time_crisis6_10=10833;

ventana=10;%tiempo en segundos ancho del bloque
samples=length(val)*ventana/time;% numero de muestras del bloque
if opcion==6
    samples=ceil(samples);
    N=ceil(length(val)/(samples));%numero de ventanas
elseif mod(length(val)/(samples),1)~=0
    samples=ceil(samples);
    N=length(val)/(samples);%numero de ventanas
elseif opcion==5
    samples=ceil(samples);
    N=ceil(length(val)/(samples));%numero de ventanas
else
    N=length(val)/(samples);%numero de ventanas
end

w=10; %10 muestras de superposicion
T=1:samples; %primer intervalo
Fsamp=256;

%definiciones para calculos
signals=1:23;
signals=signals';

fisher_alfa=zeros(23,N);
fisher_beta=zeros(23,N);
fisher_gamma=zeros(23,N);

curtosis_alfa=zeros(23,N);
curtosis_beta=zeros(23,N);
curtosis_gamma=zeros(23,N);

potencia_alfa=zeros(23,1);
potencia_beta=zeros(23,1);
potencia_gamma=zeros(23,1);

%filtros
filtro_alfa = designfilt('bandpassiir', 'StopbandFrequency1', 7.9,
'PassbandFrequency1', 8, 'PassbandFrequency2', 12.9, 'StopbandFrequency2',
13, 'StopbandAttenuation1', 30, 'PassbandRipple', 1, 'StopbandAttenuation2',
30, 'SampleRate', 256, 'DesignMethod', 'cheby1');
filtro_beta = designfilt('bandpassiir', 'StopbandFrequency1', 12.9,
'PassbandFrequency1', 13, 'PassbandFrequency2', 29.9, 'StopbandFrequency2',
30, 'StopbandAttenuation1', 30, 'PassbandRipple', 1, 'StopbandAttenuation2',
30, 'SampleRate', 256, 'DesignMethod', 'cheby1');
filtro_gamma = designfilt('bandpassiir', 'StopbandFrequency1', 29.9,
'PassbandFrequency1', 30, 'PassbandFrequency2', 60, 'StopbandFrequency2',
60.1, 'StopbandAttenuation1', 30, 'PassbandRipple', 1,
'StopbandAttenuation2', 30, 'SampleRate', 256, 'DesignMethod', 'cheby1');

toc
comp_alfa=filtfilt(filtro_alfa,val');%aplico el filtrado en freq
toc
comp_beta=filtfilt(filtro_beta,val');%aplico el filtrado en freq;
toc
comp_gamma=filtfilt(filtro_gamma,val');%aplico el filtrado en freq
toc

```

```

comp_alfa=comp_alfa';
comp_beta=comp_beta';
comp_gamma=comp_gamma';
toc

%se?al x_t primera iteracion

x_t_alfa(:,T)=comp_alfa(:,T);
x_t_beta(:,T)=comp_beta(:,T);
x_t_gamma(:,T)=comp_gamma(:,T);

%C?lculos

%coef de asimetria
fish_alfa=skewness(x_t_alfa');
fisher_alfa(signals,1)=fish_alfa;
fish_beta=skewness(x_t_beta');
fisher_beta(signals,1)=fish_beta;
fish_gamma=skewness(x_t_gamma');
fisher_gamma(signals,1)=fish_gamma;

%curtosis
curt_alfa=kurtosis(x_t_alfa');
curtosis_alfa(signals,1)=curt_alfa;
curt_beta=kurtosis(x_t_beta');
curtosis_beta(signals,1)=curt_beta;
curt_gamma=kurtosis(x_t_gamma');
curtosis_gamma(signals,1)=curt_gamma;

%potencia de cada banda

[bw_alfa,flo_alfa,fhi_alfa,powr_alfa] = obw(x_t_alfa',Fsamp);
[bw_beta,flo_beta,fhi_beta,powr_beta] = obw(x_t_beta',Fsamp);
[bw_gamma,flo_gamma,fhi_gamma,powr_gamma] = obw(x_t_gamma',Fsamp);
potencia_alfa(signals,1)=powr_alfa;
potencia_beta(signals,1)=powr_beta;
potencia_gamma(signals,1)=powr_gamma;

toc

% iteraciones siguientes
n=2;
for j=samples-w:samples:length(val)-w-1
z=j;
h=z:n*samples;

if opcion==6&&n==N
    h=h(1:601);
elseif opcion==5&&n==N
    h=h(1:2259);
end
x_t_alfa(:,h)=comp_alfa(:,h);

```

```

x_t_beta(:,h)=comp_beta(:,h);
x_t_gamma(:,h)=comp_gamma(:,h);

%Cálculos
toc

%coef de asimetria

fish_alfa=skewness(x_t_alfa');
fisher_alfa(signals,n)=fish_alfa;
fish_beta=skewness(x_t_beta');
fisher_beta(signals,n)=fish_beta;
fish_gamma=skewness(x_t_gamma');
fisher_gamma(signals,n)=fish_gamma;

%curtosis
curt_alfa=kurtosis(x_t_alfa');
curtosis_alfa(signals,n)=curt_alfa;
curt_beta=kurtosis(x_t_beta');
curtosis_beta(signals,n)=curt_beta;
curt_gamma=kurtosis(x_t_gamma');
curtosis_gamma(signals,n)=curt_gamma;

%potencia de cada banda

[bw_alfa,flo_alfa,fhi_alfa,powr_alfa] = obw(x_t_alfa',Fsamp);
[bw_beta,flo_beta,fhi_beta,powr_beta] = obw(x_t_beta',Fsamp);
[bw_gamma,flo_gamma,fhi_gamma,powr_gamma] = obw(x_t_gamma',Fsamp);
potencia_alfa(signals,n)=powr_alfa;
potencia_beta(signals,n)=powr_beta;
potencia_gamma(signals,n)=powr_gamma;

toc
n=n+1;
end

if(opcion==1&&opcion2==1||opcion==1&&opcion2==5||opcion==1&&opcion2==10||opcion==2&&opcion2==10||opcion==2&&opcion2==35||opcion==3&&opcion2==5||opcion==5&&opcion2==1||opcion==5&&opcion2==39)
    disp('este paciente no hay tenido ningun ataque epileptico')

bin0=zeros(N,1);

%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin0);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin0);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin0);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
    fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin0
);

else
    switch opcion
        case 1

```

```

switch opcion2
case 3
    %vector binario de crisis epileptica para el caso 3

    seg_pre=2000;%segundos antes del ataque que se estudiar?
samples_crisis_3=length(val)*ventana/time_crisis1_3;
N_crisis_3=ceil(length(val)/(samples_crisis_3));
bin3=zeros(N,1);

time_ini_N=time_crisis1_3-seg_pre;
samples_ini_crisis_3=length(val)*ventana/time_ini_N;
N_ini_crisis_3=ceil(length(val)/(samples_ini_crisis_3));

bin3(N_ini_crisis_3:N_crisis_3)=1;

%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin3);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin3);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin3);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin3
);

case 4
    %vector binario de crisis epileptica para el caso 4

    seg_pre=1400;%segundos antes del ataque que se estudiar?

samples_crisis_4=length(val)*ventana/time_crisis1_4;
N_crisis_4=ceil(length(val)/(samples_crisis_4));
bin4=zeros(N,1);
time_ini_N=time_crisis1_4-seg_pre;
samples_ini_crisis_4=length(val)*ventana/time_ini_N;
N_ini_crisis_4=ceil(length(val)/(samples_ini_crisis_4));

bin4(N_ini_crisis_4:N_crisis_4)=1;

%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin4);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin4);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin4);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin4
);

case 15
    %vector binario de crisis epileptica para el caso 15

```

```
seg_pre=1500;%segundos antes del ataque que se estudiar?n

samples_crisis_15=length(val)*ventana/time_crisis1_15;
N_crisis_15=ceil(length(val)/(samples_crisis_15));
bin15=zeros(N,1);

time_ini_N=time_crisis1_15-seg_pre;
samples_ini_crisis_15=length(val)*ventana/time_ini_N;
N_ini_crisis_15=ceil(length(val)/(samples_ini_crisis_15));

bin15(N_ini_crisis_15:N_crisis_15)=1;

%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin15);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin15);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin15);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
5);

otherwise
    disp(' error')
end

case 2
    switch opcion2
        case 16
            seg_pre=100;%segundos antes del ataque que se estudiar?n

            samples_crisis_16=length(val)*ventana/time_crisis2_16;
            N_crisis_16=ceil(length(val)/(samples_crisis_16));
            bin16=zeros(N,1);

            time_ini_N=time_crisis2_16-seg_pre;
            samples_ini_crisis_16=length(val)*ventana/time_ini_N;
            N_ini_crisis_16=ceil(length(val)/(samples_ini_crisis_16));

            bin16(N_ini_crisis_16:N_crisis_16)=1;

            %concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin16);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin16);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin16);
```

```

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
6);

case 17
    seg_pre=2000;%segundos antes del ataque que se
estudiar?n
        %se pueden mas

    samples_crisis_17=length(val)*ventana/time_crisis2_17;
    N_crisis_17=ceil(length(val)/(samples_crisis_17));
    bin17=zeros(N,1);

    time_ini_N=time_crisis2_17-seg_pre;
    samples_ini_crisis_17=length(val)*ventana/time_ini_N;

N_ini_crisis_17=ceil(length(val)/(samples_ini_crisis_17));

    bin17(N_ini_crisis_17:N_crisis_17)=1;

    %concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin17);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin17);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin17);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
7);

case 19
    %vector binario de crisis epileptica para el caso 19

    seg_pre=2000;%segundos antes del ataque que se estudiar?n
    %se pueden mas
    samples_crisis_19=length(val)*ventana/time_crisis2_19;
    N_crisis_19=ceil(length(val)/(samples_crisis_19));
    bin19=zeros(N,1);

    time_ini_N=time_crisis2_19-seg_pre;
    samples_ini_crisis_19=length(val)*ventana/time_ini_N;
N_ini_crisis_19=ceil(length(val)/(samples_ini_crisis_19));

    bin19(N_ini_crisis_19:N_crisis_19)=1;

    %concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin19);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin19);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin19);

```

```
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
9);

otherwise
    disp('error')
end
case 3

switch opcion2
    case 1
        seg_pre=300;%segundos antes del ataque
que se estudiar?n

        samples_crisis_1=length(val)*ventana/time_crisis3_1;
        N_crisis_1=ceil(length(val)/(samples_crisis_1));
        bin1=zeros(N,1);

        time_ini_N=time_crisis3_1-seg_pre;
        samples_ini_crisis_1=length(val)*ventana/time_ini_N;
        N_ini_crisis_1=ceil(length(val)/(samples_ini_crisis_1));

        bin1(N_ini_crisis_1:N_crisis_1)=1;

%concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin1);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin1);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin1);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
);

case 2

        seg_pre=700;%segundos antes del ataque
que se estudiar?n

        samples_crisis_2=length(val)*ventana/time_crisis3_2;
        N_crisis_2=ceil(length(val)/(samples_crisis_2));
        bin2=zeros(N,1);

        time_ini_N=time_crisis3_2-seg_pre;
        samples_ini_crisis_2=length(val)*ventana/time_ini_N;
        N_ini_crisis_2=ceil(length(val)/(samples_ini_crisis_2));

        bin2(N_ini_crisis_2:N_crisis_2)=1;

%concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin2);
```

```

banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin2);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin2);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin2
);

case 4

seg_pre=2000;%segundos antes del ataque
que se estudiar?n
%
se puede mas
samples_crisis_4=length(val)*ventana/time_crisis3_4;
N_crisis_4=ceil(length(val)/(samples_crisis_4));
bin4=zeros(N,1);

time_ini_N=time_crisis3_4-seg_pre;
samples_ini_crisis_4=length(val)*ventana/time_ini_N;
N_ini_crisis_4=ceil(length(val)/(samples_ini_crisis_4));

bin4(N_ini_crisis_4:N_crisis_4)=1;

%concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin4);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin4);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin4);
banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin4
);

case 35

seg_pre=2000;%segundos antes del ataque
que se estudiar?n
%
samples_crisis_35=length(val)*ventana/time_crisis3_35;
N_crisis_35=ceil(length(val)/(samples_crisis_35));
bin35=zeros(N,1);

time_ini_N=time_crisis3_35-seg_pre;
samples_ini_crisis_35=length(val)*ventana/time_ini_N;

N_ini_crisis_35=ceil(length(val)/(samples_ini_crisis_35));

bin35(N_ini_crisis_35:N_crisis_35)=1;

%concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin35);

```

```
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin35);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin35);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin3
5);

    otherwise
    disp('error')
end

case 5

switch opcion2

case 6
    seg_pre=400;%segundos antes del ataque que se estudiar?n

    samples_crisis_6=length(val)*ventana/time_crisis5_6;
    N_crisis_6=ceil(length(val)/(samples_crisis_6));
    bin6=zeros(N,1);

    time_ini_N=time_crisis5_6-seg_pre;
    samples_ini_crisis_6=length(val)*ventana/time_ini_N;
    N_ini_crisis_6=ceil(length(val)/(samples_ini_crisis_6));

    bin6(N_ini_crisis_6:N_crisis_6)=1;

    %concatenacion por bandas


banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin6);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin6);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin6);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin6
);

case 17
    seg_pre=2000;%segundos antes del ataque que se estudiar?n
    %se puede mas
    samples_crisis_17=length(val)*ventana/time_crisis5_17;
    N_crisis_17=ceil(length(val)/(samples_crisis_17));
    bin17=zeros(N,1);

    time_ini_N=time_crisis5_17-seg_pre;
    samples_ini_crisis_17=length(val)*ventana/time_ini_N;
    N_ini_crisis_17=ceil(length(val)/(samples_ini_crisis_17));

    bin17(N_ini_crisis_17:N_crisis_17)=1;

    %concatenacion por bandas
```

```

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin17);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin17);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin17);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
7);

case 22
    %vector binario de crisis epileptica para el caso 22

    seg_pre=2000;%segundos antes del ataque que se estudiar?n

    samples_crisis_22=length(val)*ventana/time_crisis5_22;
    N_crisis_22=ceil(length(val)/(samples_crisis_22));
    bin22=zeros(N,1);

    time_ini_N=time_crisis5_22-seg_pre;
    samples_ini_crisis_22=length(val)*ventana/time_ini_N;
    N_ini_crisis_22=ceil(length(val)/(samples_ini_crisis_22));

    bin22(N_ini_crisis_22:N_crisis_22)=1;

    %concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin22);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin22);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin22);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin2
2);

    otherwise
        disp('dato introducido erroneo')
        end

case 6
    switch opcion2
    case 9
        seg_pre=8000;%segundos antes del ataque que se estudiar?n
%se pueden muchos mas
        samples_crisis_9=length(val)*ventana/time_crisis6_9;
        N_crisis_9=ceil(length(val)/(samples_crisis_9));
        bin9=zeros(N,1);

        time_ini_N=time_crisis6_9-seg_pre;
        samples_ini_crisis_9=length(val)*ventana/time_ini_N;
        N_ini_crisis_9=ceil(length(val)/(samples_ini_crisis_9));

        bin9(N_ini_crisis_9:N_crisis_9)=1;

```

```
%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin9);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin9);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin9);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin9
);

    case 10
        seg_pre=8000;%segundos antes del ataque que se estudiar?

        samples_crisis_10=length(val)*ventana/time_crisis6_10;
        N_crisis_10=ceil(length(val)/(samples_crisis_10));
        bin10=zeros(N,1);

        time_ini_N=time_crisis6_10-seg_pre;
        samples_ini_crisis_10=length(val)*ventana/time_ini_N;
        N_ini_crisis_10=ceil(length(val)/(samples_ini_crisis_10));

        bin10(N_ini_crisis_10:N_crisis_10)=1;

%concatenacion por bandas

banda_alfa=horzcat(potencia_alfa',fisher_alfa',curtosis_alfa',bin10);
banda_beta=horzcat(potencia_beta',fisher_beta',curtosis_beta',bin10);
banda_gamma=horzcat(potencia_gamma',fisher_gamma',curtosis_gamma',bin10);

banda_tot=horzcat(potencia_alfa',potencia_beta',potencia_gamma',fisher_alfa',
fisher_beta',fisher_gamma',curtosis_alfa',curtosis_beta',curtosis_gamma',bin1
0);

    otherwise
        disp('error')
    end

otherwise
    disp('error')
end
end

pruebas;
```

7.2 Script 2: *pruebas.m*

```
%Pruebas

if(opcion==1&&opcion2==1||opcion==1&&opcion2==5||opcion==1&&opcion2==10||opcion==2&&opcion2==10||opcion==2&&opcion2==35||opcion==3&&opcion2==5||opcion==5&&opcion2==1||opcion==5&&opcion2==39)

    %SVM1 entrena con curtosis del paciente 4
    kurto1=horzcat(curtosis_alfa',bin0);
    kurto2=horzcat(curtosis_beta',bin0);
    kurto3=horzcat(curtosis_gamma',bin0);
    %SVM2 entrena con el coeficiente de asimetria del paciente 4
    asimetric1=horzcat(fisher_alfa',bin0);
    asimetric2=horzcat(fisher_beta',bin0);
    asimetric3=horzcat(fisher_gamma',bin0);
    %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
    %datos (vectores)en las bandas 100 150 200 y 300
    pot_senza1=horzcat(potencia_alfa',bin0);
    pot_senza2=horzcat(potencia_beta',bin0);
    pot_senza3=horzcat(potencia_gamma',bin0);

else
switch opcion
    case 1
        switch opcion2
            case 3
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin3);
                kurto2=horzcat(curtosis_beta',bin3);
                kurto3=horzcat(curtosis_gamma',bin3);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin3);
                asimetric2=horzcat(fisher_beta',bin3);
                asimetric3=horzcat(fisher_gamma',bin3);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin3);
                pot_senza2=horzcat(potencia_beta',bin3);
                pot_senza3=horzcat(potencia_gamma',bin3);

            case 4
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin4);
                kurto2=horzcat(curtosis_beta',bin4);
                kurto3=horzcat(curtosis_gamma',bin4);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin4);
                asimetric2=horzcat(fisher_beta',bin4);
                asimetric3=horzcat(fisher_gamma',bin4);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin4);
                pot_senza2=horzcat(potencia_beta',bin4);
                pot_senza3=horzcat(potencia_gamma',bin4);

            case 15
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin15);
```

```
kurto2=horzcat(curtosis_beta',bin15);
kurto3=horzcat(curtosis_gamma',bin15);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin15);
asimetric2=horzcat(fisher_beta',bin15);
asimetric3=horzcat(fisher_gamma',bin15);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senza1=horzcat(potencia_alfa',bin15);
pot_senza2=horzcat(potencia_beta',bin15);
pot_senza3=horzcat(potencia_gamma',bin15);

otherwise
    disp(' error')
end

case 2

    switch opcion2
        case 16
            %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin16);
kurto2=horzcat(curtosis_beta',bin16);
kurto3=horzcat(curtosis_gamma',bin16);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin16);
asimetric2=horzcat(fisher_beta',bin16);
asimetric3=horzcat(fisher_gamma',bin16);
%SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senza1=horzcat(potencia_alfa',bin16);
pot_senza2=horzcat(potencia_beta',bin16);
pot_senza3=horzcat(potencia_gamma',bin16);

        case 17

            %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin17);
kurto2=horzcat(curtosis_beta',bin17);
kurto3=horzcat(curtosis_gamma',bin17);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin17);
asimetric2=horzcat(fisher_beta',bin17);
asimetric3=horzcat(fisher_gamma',bin17);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senza1=horzcat(potencia_alfa',bin17);
pot_senza2=horzcat(potencia_beta',bin17);
pot_senza3=horzcat(potencia_gamma',bin17);

        case 19
            %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin19);
kurto2=horzcat(curtosis_beta',bin19);
kurto3=horzcat(curtosis_gamma',bin19);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin19);
asimetric2=horzcat(fisher_beta',bin19);
asimetric3=horzcat(fisher_gamma',bin19);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senza1=horzcat(potencia_alfa',bin19);
pot_senza2=horzcat(potencia_beta',bin19);
pot_senza3=horzcat(potencia_gamma',bin19);
```

```

        otherwise
            disp(' error')
        end
    case 3
        switch opcion2
            case 1
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin1);
                kurto2=horzcat(curtosis_beta',bin1);
                kurto3=horzcat(curtosis_gamma',bin1);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin1);
                asimetric2=horzcat(fisher_beta',bin1);
                asimetric3=horzcat(fisher_gamma',bin1);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin1);
                pot_senza2=horzcat(potencia_beta',bin1);
                pot_senza3=horzcat(potencia_gamma',bin1);

            case 2
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin2);
                kurto2=horzcat(curtosis_beta',bin2);
                kurto3=horzcat(curtosis_gamma',bin2);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin2);
                asimetric2=horzcat(fisher_beta',bin2);
                asimetric3=horzcat(fisher_gamma',bin2);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin2);
                pot_senza2=horzcat(potencia_beta',bin2);
                pot_senza3=horzcat(potencia_gamma',bin2);

            case 4
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin4);
                kurto2=horzcat(curtosis_beta',bin4);
                kurto3=horzcat(curtosis_gamma',bin4);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin4);
                asimetric2=horzcat(fisher_beta',bin4);
                asimetric3=horzcat(fisher_gamma',bin4);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin4);
                pot_senza2=horzcat(potencia_beta',bin4);
                pot_senza3=horzcat(potencia_gamma',bin4);

            case 35
                %SVM1 entrena con curtosis del paciente 4
                kurto1=horzcat(curtosis_alfa',bin35);
                kurto2=horzcat(curtosis_beta',bin35);
                kurto3=horzcat(curtosis_gamma',bin35);
                %SVM2 entrena con el coeficiente de asimetria del paciente 4
                asimetric1=horzcat(fisher_alfa',bin35);
                asimetric2=horzcat(fisher_beta',bin35);
                asimetric3=horzcat(fisher_gamma',bin35);
                %SVM3 entrena con la potencia en banda alfa del paciente 3 con menos
                %datos (vectores)en las bandas 100 150 200 y 300
                pot_senza1=horzcat(potencia_alfa',bin35);
                pot_senza2=horzcat(potencia_beta',bin35);
                pot_senza3=horzcat(potencia_gamma',bin35);

```

```
        otherwise
            disp(' error')
    end
case 5
    switch opcion2
    case 6
        %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin6);
kurto2=horzcat(curtosis_beta',bin6);
kurto3=horzcat(curtosis_gamma',bin6);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin6);
asimetric2=horzcat(fisher_beta',bin6);
asimetric3=horzcat(fisher_gamma',bin6);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senzal=horzcat(potencia_alfa',bin6);
pot_senza2=horzcat(potencia_beta',bin6);
pot_senza3=horzcat(potencia_gamma',bin6);
    case 17
        %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin17);
kurto2=horzcat(curtosis_beta',bin17);
kurto3=horzcat(curtosis_gamma',bin17);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin17);
asimetric2=horzcat(fisher_beta',bin17);
asimetric3=horzcat(fisher_gamma',bin17);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senzal=horzcat(potencia_alfa',bin17);
pot_senza2=horzcat(potencia_beta',bin17);
pot_senza3=horzcat(potencia_gamma',bin17);
    case 22
        %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin22);
kurto2=horzcat(curtosis_beta',bin22);
kurto3=horzcat(curtosis_gamma',bin22);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin22);
asimetric2=horzcat(fisher_beta',bin22);
asimetric3=horzcat(fisher_gamma',bin22);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senzal=horzcat(potencia_alfa',bin22);
pot_senza2=horzcat(potencia_beta',bin22);
pot_senza3=horzcat(potencia_gamma',bin22);
        otherwise
            disp(' error')
    end
case 6
    switch opcion2
    case 9
        %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin9);
kurto2=horzcat(curtosis_beta',bin9);
kurto3=horzcat(curtosis_gamma',bin9);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin9);
asimetric2=horzcat(fisher_beta',bin9);
asimetric3=horzcat(fisher_gamma',bin9);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senzal=horzcat(potencia_alfa',bin9);
pot_senza2=horzcat(potencia_beta',bin9);
```

```

pot_senza3=horzcat(potencia_gamma',bin9);
    case 10
        %SVM1 entrena con curtosis del paciente 4
kurto1=horzcat(curtosis_alfa',bin10);
kurto2=horzcat(curtosis_beta',bin10);
kurto3=horzcat(curtosis_gamma',bin10);
%SVM2 entrena con el coeficiente de asimetria del paciente 4
asimetric1=horzcat(fisher_alfa',bin10);
asimetric2=horzcat(fisher_beta',bin10);
asimetric3=horzcat(fisher_gamma',bin10);
%SVM2 entrena con la potencia en banda alfa del paciente 3 con menos
%datos (vectores)en las bandas 100 150 200 y 300
pot_senza1=horzcat(potencia_alfa',bin10);
pot_senza2=horzcat(potencia_beta',bin10);
pot_senza3=horzcat(potencia_gamma',bin10);
    otherwise
        disp(' error')
    end

otherwise
    disp('error')
end
end

%PRUEBAS
disp('Curtosis')
%CON SVM1 - curtosis
[tcc1,vcc1]=kur(kurto1);
figure
subplot(3,1,1),plot(tcc1.predictFcn(kurto1(:,1:23)))
title('SVM= Curtosis Datos= Curtosis banda alfa')
hold on
%probabilidad de crisis
vector=tcc1.predictFcn(kurto1(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
    no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_no_crisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
curtosis y entrenandolo con curtosis alfa', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM
curtosis y entrenandolo con curtosis alfa', prob_no_crisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
curtosis y entrenandolo con curtosis alfa')
end

```

```
[tcc2,vcc2]=kur(kurto2);
subplot(3,1,2),plot(tcc2.predictFcn(kurto2(:,1:23)))
title('SVM= Curtosis Datos= Curtosis banda beta')
hold on
%probabilidad de crisis
vector=tcc2.predictFcn(kurto2(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM curtosis y entrenandolo con curtosis beta', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM curtosis y entrenandolo con curtosis beta', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM curtosis y entrenandolo con curtosis beta')
end

[tcc3,vcc3]=kur(kurto3);
subplot(3,1,3),plot(tcc3.predictFcn(kurto3(:,1:23)))
title('SVM= Curtosis Datos= Curtosis banda gamma')
hold on
%probabilidad de crisis
vector=tcc3.predictFcn(kurto3(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM curtosis y entrenandolo con curtosis gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
```

```

elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM curtosis y entrenandolo con curtosis gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
curtosis y entrenandolo con curtosis gamma')
end
%
hold off

%CON SVM2 - Coef de asimetria
disp('Coeficiente de asimetria')
figure

[taa1,vaal]=asimetria(asimetric1);
subplot(3,1,1),plot(taa1.predictFcn(asimetric1(:,1:23)))
title('SVM= Coef. de Asimetria Datos= Coef. de Asimetria alfa')
hold on
%probabilidad de crisis
vector=taa1.predictFcn(asimetric1(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
alfa', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en
banda alfa', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
alfa')
end

[taa2,vaa2]=asimetria(asimetric2);
subplot(3,1,2),plot(taa2.predictFcn(asimetric2(:,1:23)))
title('SVM= Coef. de Asimetria Datos= Coef. de Asimetria beta')
hold on
%probabilidad de crisis
vector=taa2.predictFcn(asimetric2(:,1:23));

```

```
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
beta', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en
banda beta', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
beta')
end

[taa3,vaa3]=asimetria(asimetric3);
subplot(3,1,3),plot(taa3.predictFcn(asimetric3(:,1:23)))
title('SVM= Coef. de Asimetria Datos= Coef. de Asimetria gamma')
%probabilidad de crisis
vector=taa3.predictFcn(asimetric3(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM Coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en
```

```

banda gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
coeficiente de asimetria y entrenandolo con Coeficiente de asimetria en banda
gamma')
end

hold off

%CON SVM4 - potencia alfa limitada
disp('Potencia alfa')
figure
[tpap1, vpap1]=pot_asin(pot_senzal);
subplot(3,3,1), plot(tpap1.predictFcn(pot_senzal(:,1:23)))
title('SVM= Potencia alfa s Datos= Potencia alfa')
hold on
%probabilidad de crisis
vector=tpap1.predictFcn(pot_senzal(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
alfa', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda alfa con informacion limitada y entrenandolo con
potencia alfa', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
alfa')
end

[tpap2, vpap2]=pot_asin(pot_senza2);
subplot(3,3,2), plot(tpap2.predictFcn(pot_senza2(:,1:23)))
title('SVM= Potencia alfa s Datos= Potencia beta')
hold on
%probabilidad de crisis
vector=tpap2.predictFcn(pot_senza2(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la

```

```
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
beta', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con
potencia beta', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
beta')
end

[tpap3,vpap3]=pot_asin(pot_senza3);
subplot(3,3,3),plot(tpap3.predictFcn(pot_senza3(:,1:23)))
title('SVM= Potencia alfa s Datos= Potencia gamma')
%probabilidad de crisis
vector=tpap3.predictFcn(pot_senza3(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con
potencia gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
```

```

disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda alfa con informacion limitada y entrenandolo con potencia
gamma')
end
hold off

%CON SVM4 - potencia beta limitada
disp('Potencia beta')
[tpbp1,vpbp1]=pot_bsin(pot_senza1);
subplot(3,3,4),plot(tpbp1.predictFcn(pot_senza1(:,1:23)))
title('SVM= Potencia beta s Datos= Potencia alfa')
hold on
%probabilidad de crisis
vector=tpbp1.predictFcn(pot_senza1(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
alfa', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda beta con informacion limitada y entrenandolo con
potencia alfa', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
alfa')
end

[tpbp2,vpbp2]=pot_bsin(pot_senza2);
subplot(3,3,5),plot(tpbp2.predictFcn(pot_senza2(:,1:23)))
title('SVM= Potencia beta s Datos= Potencia beta')
hold on
%probabilidad de crisis
vector=tpbp2.predictFcn(pot_senza2(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end

```

```
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
beta', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
beta', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
beta')
end
[tpbp3,vpbp3]=pot_bsin(pot_senza3);
subplot(3,3,6),plot(tpbp3.predictFcn(pot_senza3(:,1:23)))
title('SVM= Potencia beta s Datos= Potencia gamma')

%probabilidad de crisis
vector=tpbp3.predictFcn(pot_senza3(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda beta con informacion limitada y entrenandolo con potencia
gamma')
end
hold off
```

```
%CON SVM5 - potencia gamma limitada
disp('Potencia gamma')

[tpgp1,vpgp1]=pot_gsin(pot_senza1);
subplot(3,3,7),plot(tpgp1.predictFcn(pot_senza1(:,1:23)))
title('SVM= Potencia gamma s Datos= Potencia alfa')
hold on
%probabilidad de crisis
vector=tpgp1.predictFcn(pot_senza1(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
alfa', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda gamma con informacion limitada y entrenandolo con
potencia alfa', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
alfa')
end

[tpgp2,vpgp2]=pot_gsin(pot_senza2);
subplot(3,3,8),plot(tpgp2.predictFcn(pot_senza2(:,1:23)))
title('SVM= Potencia gamma s Datos= Potencia beta')
hold on
%probabilidad de crisis
vector=tpgp2.predictFcn(pot_senza2(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
```

```
prob_nocrisis=(no_crisis/dim_vector(1))*100;
% disp('La probabilidad de que haya una crisis es %s', prob_crisis)
if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
beta', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda gamma con informacion limitada y entrenandolo con
potencia beta', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
beta')
end

[tpgp3,vpgp3]=pot_gsin(pot_senza3);
subplot(3,3,9),plot(tpgp3.predictFcn(pot_senza3(:,1:23)))
title('SVM= Potencia gamma s Datos= Potencia gamma')

%probabilidad de crisis
vector=tpgp3.predictFcn(pot_senza3(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda gamma con informacion limitada y entrenandolo con
potencia gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
gamma')
end

hold off
disp('temprana')
```

```

[tpgp4,vpgp4]=temprano(pot_senzal);
subplot(3,3,9),plot(tpgp4.predictFcn(pot_senzal(:,1:23)))
title('SVM= Potencia gamma temprana Datos= Potencia gamma')

%probabilidad de crisis
vector=tpgp4.predictFcn(pot_senzal(:,1:23));
dim_vector=size(vector);
crisis=0;
no_crisis=0;
for i=1:dim_vector(1) %contador de crisis o no crisis para calcular la
probabilidad
if vector(i,1)==1
    crisis=crisis+1;
else
no_crisis=no_crisis+1;
end
end

prob_crisis= (crisis/dim_vector(1))*100;
prob_nocrisis=(no_crisis/dim_vector(1))*100;

if crisis > no_crisis
fprintf('Hay crisis con una probabilidad de %02f por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
gamma', prob_crisis)
%Muestra la probabilidad de crisis si es mayor de 50%
disp('%')
elseif no_crisis > crisis

fprintf('No hay crisis con una probabilidad de %02f por ciento usando como
SVM potencia en banda gamma con informacion limitada y entrenandolo con
potencia gamma', prob_nocrisis)
%Muestra la probabilidad de no crisis si es mayor de 50%
disp('%')
else
disp('Misma probabilidad de crisis que no crisis por ciento usando como SVM
potencia en banda gamma con informacion limitada y entrenandolo con potencia
gamma')
end

hold off

```

7.3 Script 3: MSVc (*kur.m*)

```

function [trainedClassifier, validationAccuracy] = kur(trainingData)
% trainClassifier(trainingData)
% returns a trained classifier and its validation accuracy.
% This code recreates the classification model trained in
% Classification Learner app.
%
% Input:
%     trainingData: the training data of same data type as imported
%                   in the app (table or matrix).
%
% Output:
%     trainedClassifier: a struct containing the trained classifier.

```

```

%
% The struct contains various fields with information about the
% trained classifier.
%
%
% trainedClassifier.predictFcn: a function to make predictions
% on new data. It takes an input of the same form as this training
% code (table or matrix) and returns predictions for the response.
% If you supply a matrix, include only the predictors columns (or
% rows).
%
%
% validationAccuracy: a double containing the validation accuracy
% score in percent. In the app, the History list displays this
% overall accuracy score for each model.
%
%
% Use the code to train the model with new data.
% To retrain your classifier, call the function from the command line
% with your original data or new data as the input argument trainingData.
%
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
% [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
%
% To make predictions with the returned 'trainedClassifier' on new data T,
% use
% yfit = trainedClassifier.predictFcn(T)
%
%
% To automate training the same classifier with new data, or to learn how
% to programmatically train classifiers, examine the generated code.
%
%
% Auto-generated by MATLAB on 05-Aug-2017 19:04:12

%
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

%
% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24')];

%
% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

%
% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitcsvm(
    predictors, ...
    response, ...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 2, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...

```

```

'Standardize', true, ...
'ClassNames', [0; 1]);

trainedClassifier.ClassificationSVM = classificationSVM;
convertMatrixToTableFcn = @(x) table(x, 'VariableNames', {'column'});
splitMatricesInTableFcn = @(t) [t(:,setdiff(t.Properties.VariableNames, ...
{'column'})), array2table(table2array(t(:,{'column'}))), 'VariableNames',
{'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6',
'column_7', 'column_8', 'column_9', 'column_10', 'column_11', 'column_12',
'column_13', 'column_14', 'column_15', 'column_16', 'column_17', 'column_18',
'column_19', 'column_20', 'column_21', 'column_22', 'column_23'})];
extractPredictorsFromTableFcn = @(t) t(:, predictorNames);
predictorExtractionFcn = @(x)
extractPredictorsFromTableFcn(splitMatricesInTableFcn(convertMatrixToTableFcn
(x)));
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames, ...
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'})];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold',
15);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

% Compute validation predictions and scores
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

```

7.4 Script 4: MSVa (asimetria.m)

```

function [trainedClassifier, validationAccuracy] = asimetria(trainingData)
% trainClassifier(trainingData)
% returns a trained classifier and its validation accuracy.
% This code recreates the classification model trained in
% Classification Learner app.

```

```

%
% Input:
%     trainingData: the training data of same data type as imported
%     in the app (table or matrix).
%
% Output:
%     trainedClassifier: a struct containing the trained classifier.
%     The struct contains various fields with information about the
%     trained classifier.
%
%     trainedClassifier.predictFcn: a function to make predictions
%     on new data. It takes an input of the same form as this training
%     code (table or matrix) and returns predictions for the response.
%     If you supply a matrix, include only the predictors columns (or
%     rows).
%
% validationAccuracy: a double containing the validation accuracy
% score in percent. In the app, the History list displays this
% overall accuracy score for each model.
%
% Use the code to train the model with new data.
% To retrain your classifier, call the function from the command line
% with your original data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
% [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data T,
% use
% yfit = trainedClassifier.predictFcn(T)
%
% To automate training the same classifier with new data, or to learn how
% to programmatically train classifiers, examine the generated code.

% Auto-generated by MATLAB on 04-Aug-2017 21:01:25

%
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

%
% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'}]];

%
% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

%
% Train a classifier
% This code specifies all the classifier options and trains the classifier.

```

```

classificationSVM = fitcsvm(
    predictors, ...
    response, ...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 3, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

trainedClassifier.ClassificationSVM = classificationSVM;
convertMatrixToTableFcn = @(x) table(x, 'VariableNames', {'column'});
splitMatricesInTableFcn = @(t) [t(:,setdiff(t.Properties.VariableNames,
{'column'})), array2table(table2array(t(:,{'column'}))), 'VariableNames',
{'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6',
'column_7', 'column_8', 'column_9', 'column_10', 'column_11', 'column_12',
'column_13', 'column_14', 'column_15', 'column_16', 'column_17', 'column_18',
'column_19', 'column_20', 'column_21', 'column_22', 'column_23'})];
extractPredictorsFromTableFcn = @(t) t(:, predictorNames);
predictorExtractionFcn = @(x)
extractPredictorsFromTableFcn(splitMatricesInTableFcn(convertMatrixToTableFcn
(x)));
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24')];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold',
20);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

% Compute validation predictions and scores
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

```

7.5 Script 5: MSVpa (*pot_asin.m*)

```

function [trainedClassifier, validationAccuracy] = pot_asin(trainingData)
% trainClassifier(trainingData)
% returns a trained classifier and its validation accuracy.
% This code recreates the classification model trained in
% Classification Learner app.
%
% Input:
%     trainingData: the training data of same data type as imported
%     in the app (table or matrix).
%
% Output:
%     trainedClassifier: a struct containing the trained classifier.
%     The struct contains various fields with information about the
%     trained classifier.
%
%     trainedClassifier.predictFcn: a function to make predictions
%     on new data. It takes an input of the same form as this training
%     code (table or matrix) and returns predictions for the response.
%     If you supply a matrix, include only the predictors columns (or
%     rows).
%
%     validationAccuracy: a double containing the validation accuracy
%     score in percent. In the app, the History list displays this
%     overall accuracy score for each model.
%
% Use the code to train the model with new data.
% To retrain your classifier, call the function from the command line
% with your original data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
% [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data T,
% use
% yfit = trainedClassifier.predictFcn(T)
%
% To automate training the same classifier with new data, or to learn how
% to programmatically train classifiers, examine the generated code.

% Auto-generated by MATLAB on 06-Aug-2017 16:20:50

%
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

%
% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'})];

%
% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',

```

```

'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitcsvm(...,
    predictors, ...
    response, ...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 2, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

trainedClassifier.ClassificationSVM = classificationSVM;
convertMatrixToTableFcn = @(x) table(x, 'VariableNames', {'column'});
splitMatricesInTableFcn = @(t) [t(:, setdiff(t.Properties.VariableNames, ...
{'column'})), array2table(table2array(t(:, {'column'}))), 'VariableNames',
{'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6',
'column_7', 'column_8', 'column_9', 'column_10', 'column_11', 'column_12',
'column_13', 'column_14', 'column_15', 'column_16', 'column_17', 'column_18',
'column_19', 'column_20', 'column_21', 'column_22', 'column_23'})];
extractPredictorsFromTableFcn = @(t) t(:, predictorNames);
predictorExtractionFcn = @(x)
extractPredictorsFromTableFcn(splitMatricesInTableFcn(convertMatrixToTableFcn
(x)));
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

% Split matrices in the input table into vectors
inputTable = [inputTable(:, setdiff(inputTable.Properties.VariableNames, ...
{'column'})), array2table(table2array(inputTable(:, {'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'})];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold',
20);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',

```

```
'ClassifError');

% Compute validation predictions and scores
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
```

7.6 Script 6: MSVpb (pot_bsin.m)

```
function [trainedClassifier, validationAccuracy] = pot_bsin(trainingData)
% trainClassifier(trainingData)
% returns a trained classifier and its validation accuracy.
% This code recreates the classification model trained in
% Classification Learner app.
%
% Input:
%   trainingData: the training data of same data type as imported
%   in the app (table or matrix).
%
% Output:
%   trainedClassifier: a struct containing the trained classifier.
%   The struct contains various fields with information about the
%   trained classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions
%   on new data. It takes an input of the same form as this training
%   code (table or matrix) and returns predictions for the response.
%   If you supply a matrix, include only the predictors columns (or
%   rows).
%
%   validationAccuracy: a double containing the validation accuracy
%   score in percent. In the app, the History list displays this
%   overall accuracy score for each model.
%
% Use the code to train the model with new data.
% To retrain your classifier, call the function from the command line
% with your original data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
%   [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data T,
% use
%   yfit = trainedClassifier.predictFcn(T)
%
% To automate training the same classifier with new data, or to learn how
% to programmatically train classifiers, examine the generated code.

% Auto-generated by MATLAB on 16-Sep-2017 21:02:23

%
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

%
% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23'}];
```

```
'column_24'])];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitcsvm(...,
    predictors, ...
    response, ...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 2, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

trainedClassifier.ClassificationSVM = classificationSVM;
convertMatrixToTableFcn = @(x) table(x, 'VariableNames', {'column'});
splitMatricesInTableFcn = @(t) [t(:, setdiff(t.Properties.VariableNames, ...
{'column'})), array2table(table2array(t(:, {'column'}))), 'VariableNames',
{'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6',
'column_7', 'column_8', 'column_9', 'column_10', 'column_11', 'column_12',
'column_13', 'column_14', 'column_15', 'column_16', 'column_17', 'column_18',
'column_19', 'column_20', 'column_21', 'column_22', 'column_23'}];
extractPredictorsFromTableFcn = @(t) t(:, predictorNames);
predictorExtractionFcn = @(x)
extractPredictorsFromTableFcn(splitMatricesInTableFcn(convertMatrixToTableFcn
(x)));
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

% Split matrices in the input table into vectors
inputTable = [inputTable(:, setdiff(inputTable.Properties.VariableNames, ...
{'column'})), array2table(table2array(inputTable(:, {'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24')];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;
```

```
% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold',
20);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

% Compute validation predictions and scores
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
```

7.7 Script 7: MSVpg (pot_gsin.m)

```
function [trainedClassifier, validationAccuracy] = pot_gsin(trainingData)
% trainClassifier(trainingData)
% returns a trained classifier and its validation accuracy.
% This code recreates the classification model trained in
% Classification Learner app.
%
% Input:
%   trainingData: the training data of same data type as imported
%   in the app (table or matrix).
%
% Output:
%   trainedClassifier: a struct containing the trained classifier.
%   The struct contains various fields with information about the
%   trained classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions
%   on new data. It takes an input of the same form as this training
%   code (table or matrix) and returns predictions for the response.
%   If you supply a matrix, include only the predictors columns (or
%   rows).
%
%   validationAccuracy: a double containing the validation accuracy
%   score in percent. In the app, the History list displays this
%   overall accuracy score for each model.
%
% Use the code to train the model with new data.
% To retrain your classifier, call the function from the command line
% with your original data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
% [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data T,
% use
% yfit = trainedClassifier.predictFcn(T)
%
% To automate training the same classifier with new data, or to learn how
% to programmatically train classifiers, examine the generated code.

% Auto-generated by MATLAB on 16-Sep-2017 21:04:13

%
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};
```

```
% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'})];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitcsvm(...%
    predictors, ...
    response, ...
    'KernelFunction', 'polynomial', ...
    'PolynomialOrder', 2, ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

trainedClassifier.ClassificationSVM = classificationSVM;
convertMatrixToTableFcn = @(x) table(x, 'VariableNames', {'column'});
splitMatricesInTableFcn = @(t) [t(:,setdiff(t.Properties.VariableNames,
{'column'})), array2table(table2array(t(:,{'column'}))), 'VariableNames',
{'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6',
'column_7', 'column_8', 'column_9', 'column_10', 'column_11', 'column_12',
'column_13', 'column_14', 'column_15', 'column_16', 'column_17', 'column_18',
'column_19', 'column_20', 'column_21', 'column_22', 'column_23'}];
extractPredictorsFromTableFcn = @(t) t(:, predictorNames);
predictorExtractionFcn = @(x)
extractPredictorsFromTableFcn(splitMatricesInTableFcn(convertMatrixToTableFcn
(x)));
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
% Convert input to table
inputTable = table(trainingData);
inputTable.Properties.VariableNames = {'column'};

% Split matrices in the input table into vectors
inputTable = [inputTable(:,setdiff(inputTable.Properties.VariableNames,
{'column'})), array2table(table2array(inputTable(:,{'column'}))),
'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22', 'column_23',
'column_24'})];

% Extract predictors and response
% This code processes the data into the right shape for training the
% classifier.
predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5',
```

```
'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20', 'column_21', 'column_22',
'column_23'});
predictors = inputTable(:, predictorNames);
response = inputTable.column_24;

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold',
20);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

% Compute validation predictions and scores
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
```

