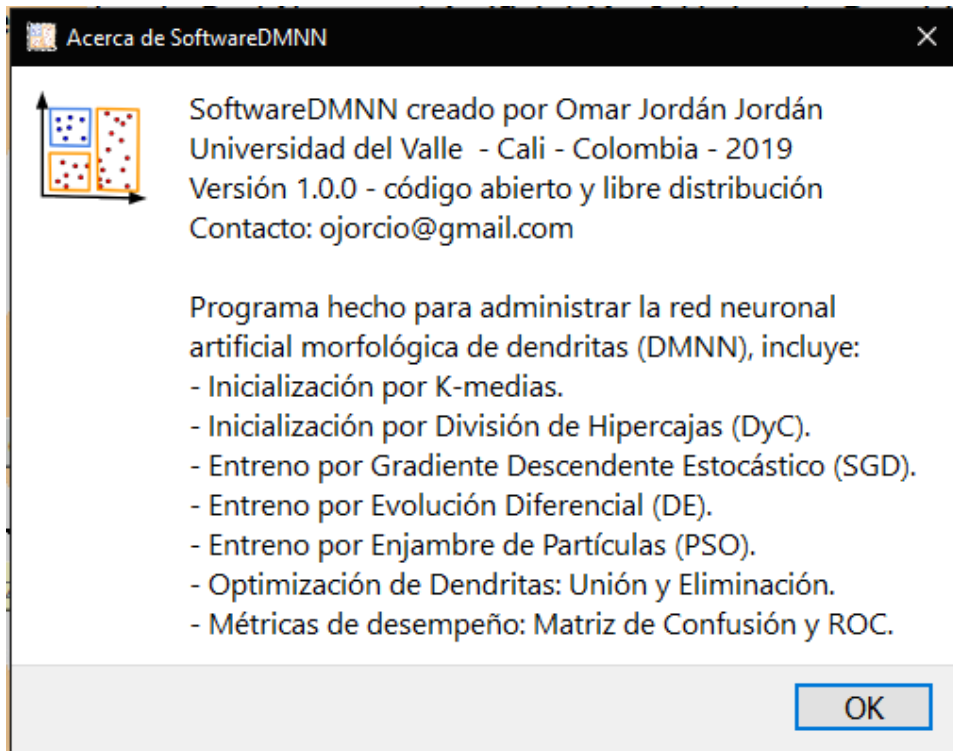
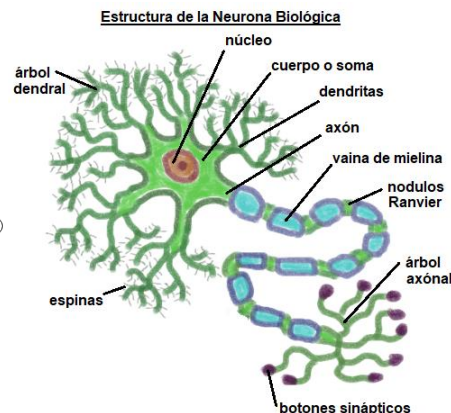
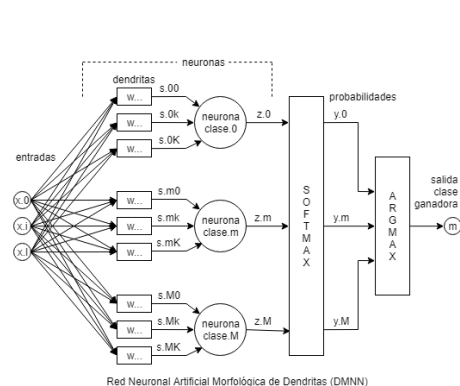


GUÍA TUTORIAL DEL SOFTWARE PARA TESTEAR LA DMNN



Omar Jordán Jordán
Universidad del Valle 2019

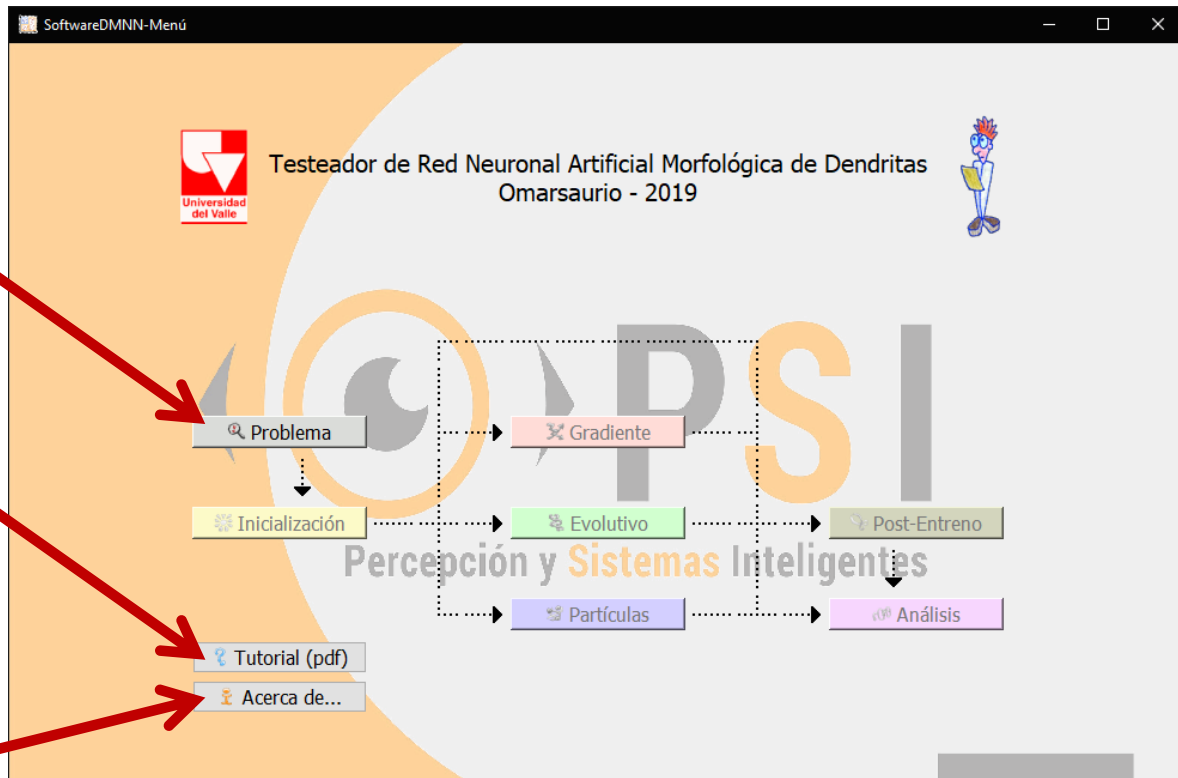


GUI - Menú

Pulse
“Problema” para
cargar datos

Muestra
esta guía

Muestra la
información
sobre el
software



Esta GUI
permite
moverse a
través del
software,
debe cargar
datos antes
de poder
crear y
entrenar la
DMNN

GUI - Problema

Esta GUI administra el set de patrones, en otras palabras define al problema de clasificación de ahí su nombre

SoftwareDMNN-Problema

Eje X . . . Eje Y

(• Ent, ■ Val, ○ Test)

Volver al menú Problema

Archivos

Importar Patrones

Manipulación de Patrones

80.00 % Patrones Entreno

15.00 % Patrones Validación

128 Bache Entreno

1024 Bache Validación

Mezclar y Exportar Calcular Porcentajes

Edición de Etiquetas

Pre-procesamiento

Min-Max Relativa

Normalizar

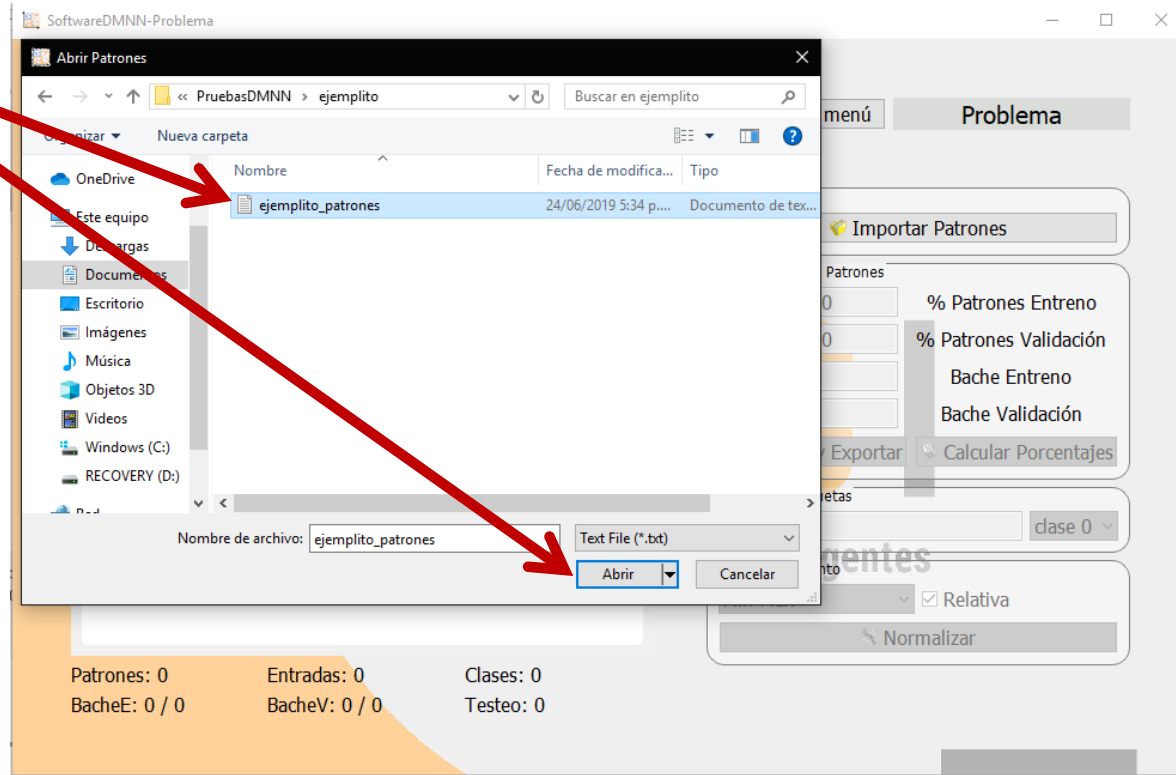
Patrones: 0 Entradas: 0 Clases: 0

BacheE: 0 / 0 BacheV: 0 / 0 Testeo: 0

Pulse “Importar Patrones” para cargar datos

GUI - Problema

Seleccione su
archivo, puede
ser TXT o XML



En caso de ser
inválida la
información,
simplemente no
se cargará

Formato Patrones TXT

El texto “Patrones: “
es obligatorio y debe
ir en la primera línea
del archivo, le sigue
el título del problema

Los textos “Salidas: “
y “Entradas: “ son
obligatorios y deben
ir en la 2da y 3ra
línea; pero luego de
sus dos puntos las
etiquetas son
opcionales

```
Patrones: TiposDeRanas
Salidas: RanaAgridulce, RanaMaliciosa
Entradas: Longitud(cm), Grosor(cm), Peso(g)
4.5, 6.2, 9.8, 0
4.7, 5.8, 6.9, 1
5.1, 5.9, 7.7, 0
4.2, 6.0, 7.0, 1
```

A partir de la 4ta fila va la matriz de
valores, cada fila es un patrón, las
primeras columnas son las entradas
y la última (derecha) es la salida

En las 3 especificaciones de
arriba, tenga en cuenta el
espacio luego de los dos puntos

En el ejemplo, hay 4 patrones,
dimensionalidad de entrada 3 y dos
salidas / clases (0 y 1)

Formato Patrones XML

Datos que definen al problema

Etiquetas de entrada y salida

Todos los datos como tal, ésta parte es fácilmente legible por Excel

```
<Patrones>
  <Titulo>TiposDeRanas</Titulo>
  <Dimension>
    <Entradas>3</Entradas>
    <Clases>2</Clases>
    <TotalPatrones>4</TotalPatrones>
  </Dimension>
  <NombresSalidas>
    <A0>RanaAgridulce</A0>
    <A1>RanaMaliciosa</A1>
  </NombresSalidas>
  <NombresEntradas>
    <N0>Longitud(cm)</N0>
    <N1>Grosor(cm)</N1>
    <N2>Peso(g)</N2>
  </NombresEntradas>
  <ValoresDatos>
    <P0><E0>4.5</E0><E1>6.2</E1><E2>9.8</E2><S>0</S></P0>
    <P1><E0>4.2</E0><E1>6.0</E1><E2>7.0</E2><S>1</S></P1>
    <P2><E0>5.1</E0><E1>5.9</E1><E2>7.7</E2><S>0</S></P2>
    <P3><E0>4.7</E0><E1>5.8</E1><E2>6.9</E2><S>1</S></P3>
  </ValoresDatos>
</Patrones>
```

```
<ParticionPatrones>
  <Entreno>2</Entreno>
  <Validacion>1</Validacion>
  <Testeo>1</Testeo>
</ParticionPatrones>
<ParticionBatches>
  <Entreno>2</Entreno>
  <Validacion>1</Validacion>
</ParticionBatches>
</Patrones>
```

Número de datos que serán de entreno, validación, testeo, bache de entreno y bache de validación

Este formato claramente abarca más información que el TXT, ahora veremos que puede ser generado por el software, se recomienda su uso

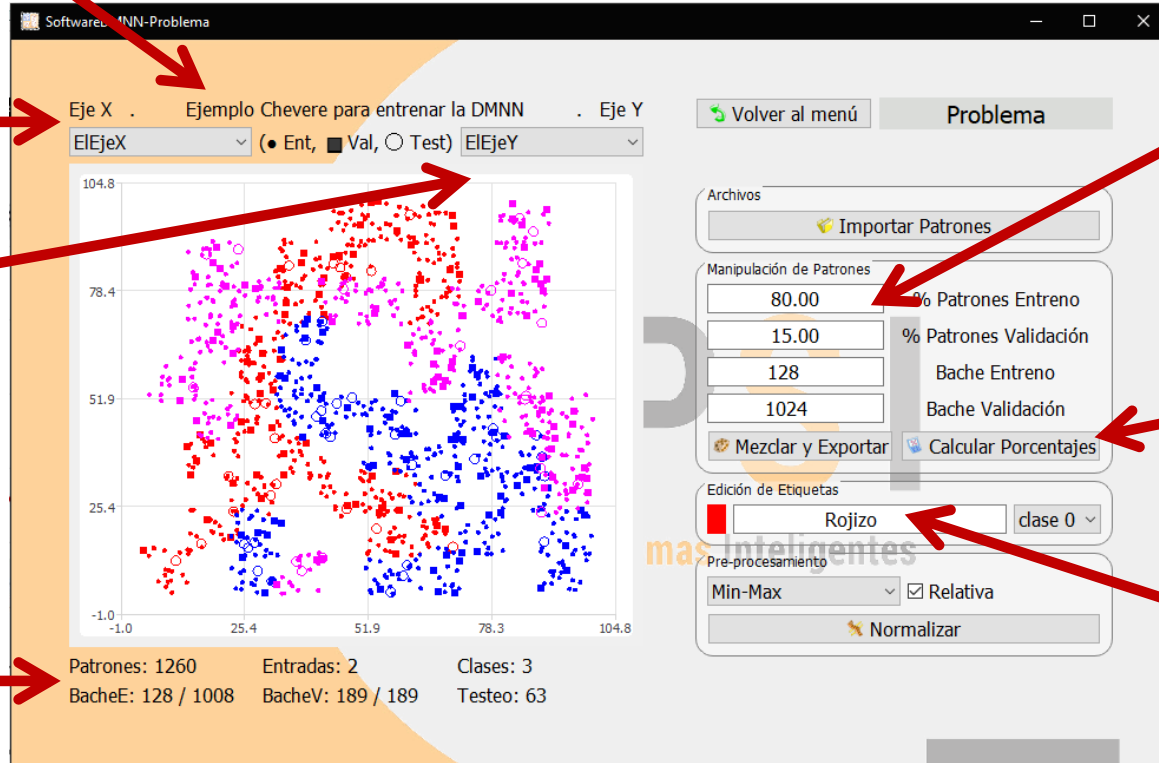
Los nombres de <elementos> No pueden cambiarse

GUI - Problema

Título del problema

Para problemas de mas de 2 dimensiones de entrada, puede seleccionar cuales va a graficar

Aquí puede ver como se distribuyen actualmente los sets



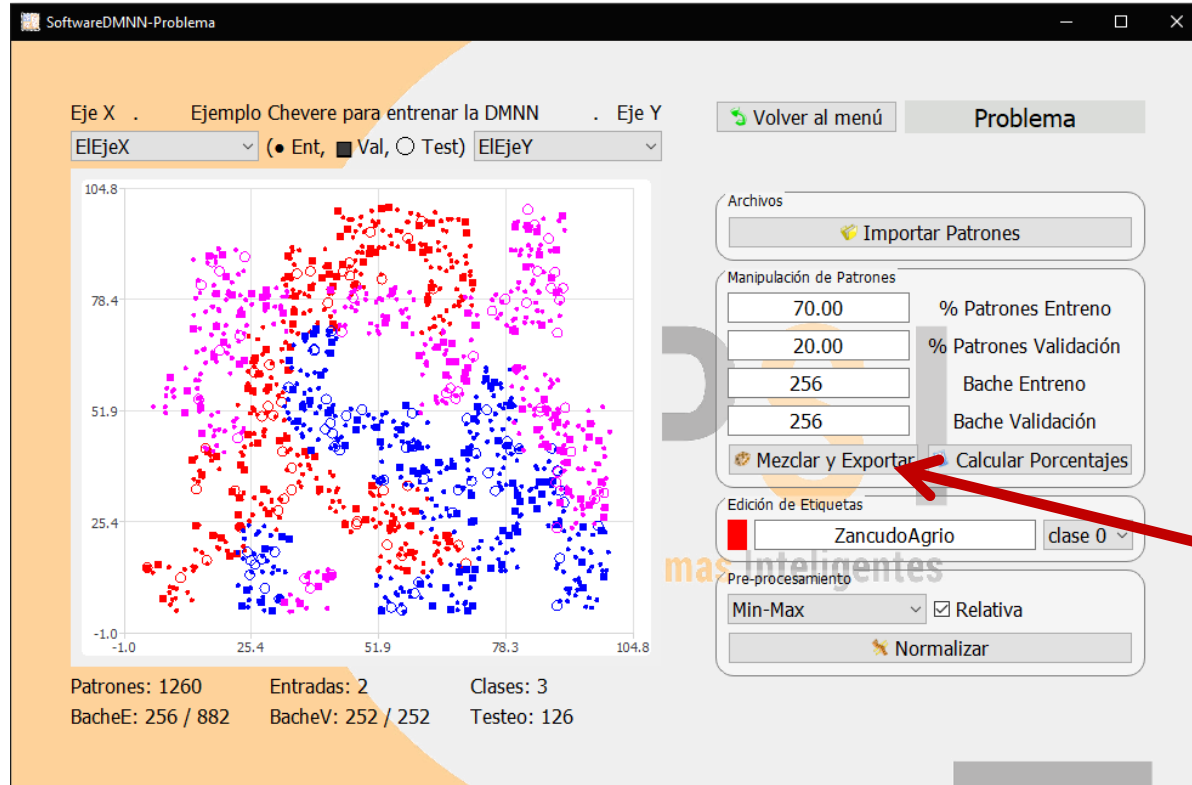
Primero digite los valores porcentuales para dividir los sets, luego los tamaños de los baches

Luego pulse "Calcular Porcentajes"

Puede editar la etiqueta / nombre de las clases

GUI - Problema

Puede observar que los puntos se han reacomodado por así decirlo

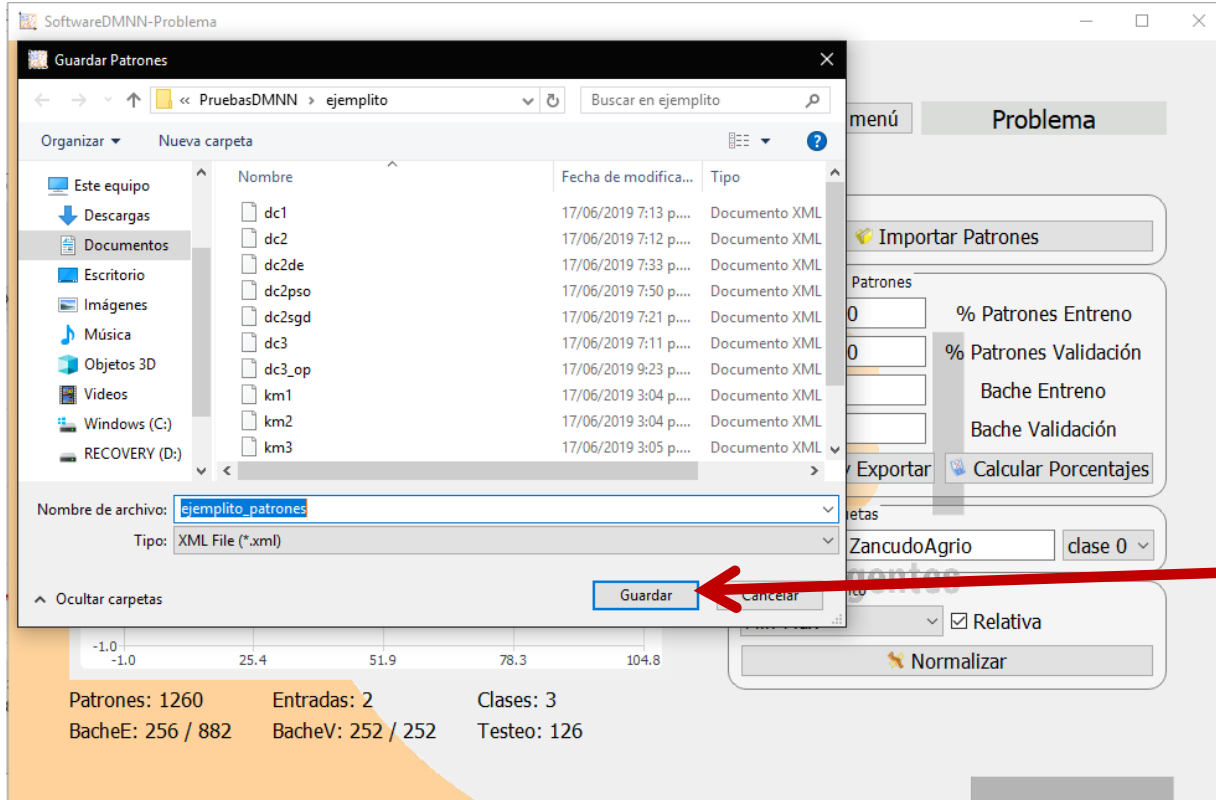


Pero a veces es necesario mezclar los patrones, más aún si el archivo de origen tiene las clases en orden

Pulse "Mezclar y Exportar"

GUI - Problema

Con el fin de poder replicar resultados en próximas ejecuciones, querrá tener el archivo de patrones con el nuevo orden

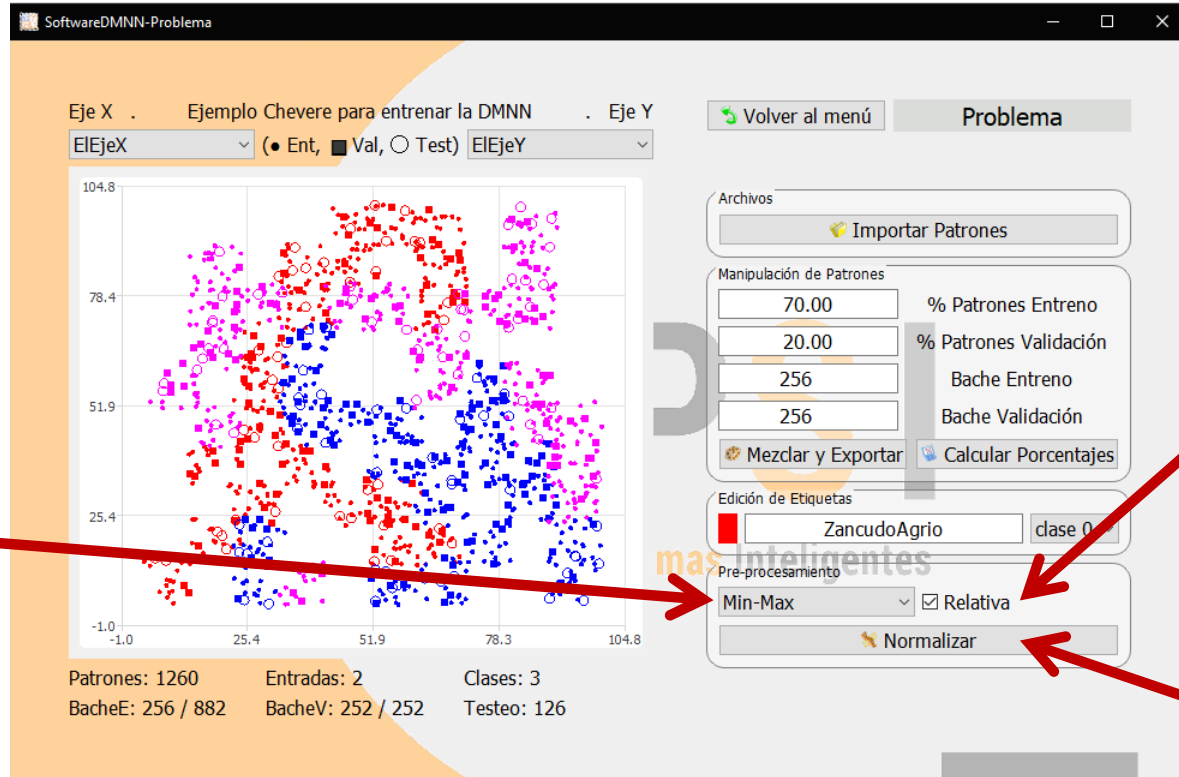


Seleccione su archivo XML y guarde, aunque este paso es opcional, ya el set se ha mezclado

GUI - Problema

Cuando se trabaja con sets de datos suele aplicarse pre procesamiento de los mismos, entre ello está la normalización, sobra decir que esto es opcional

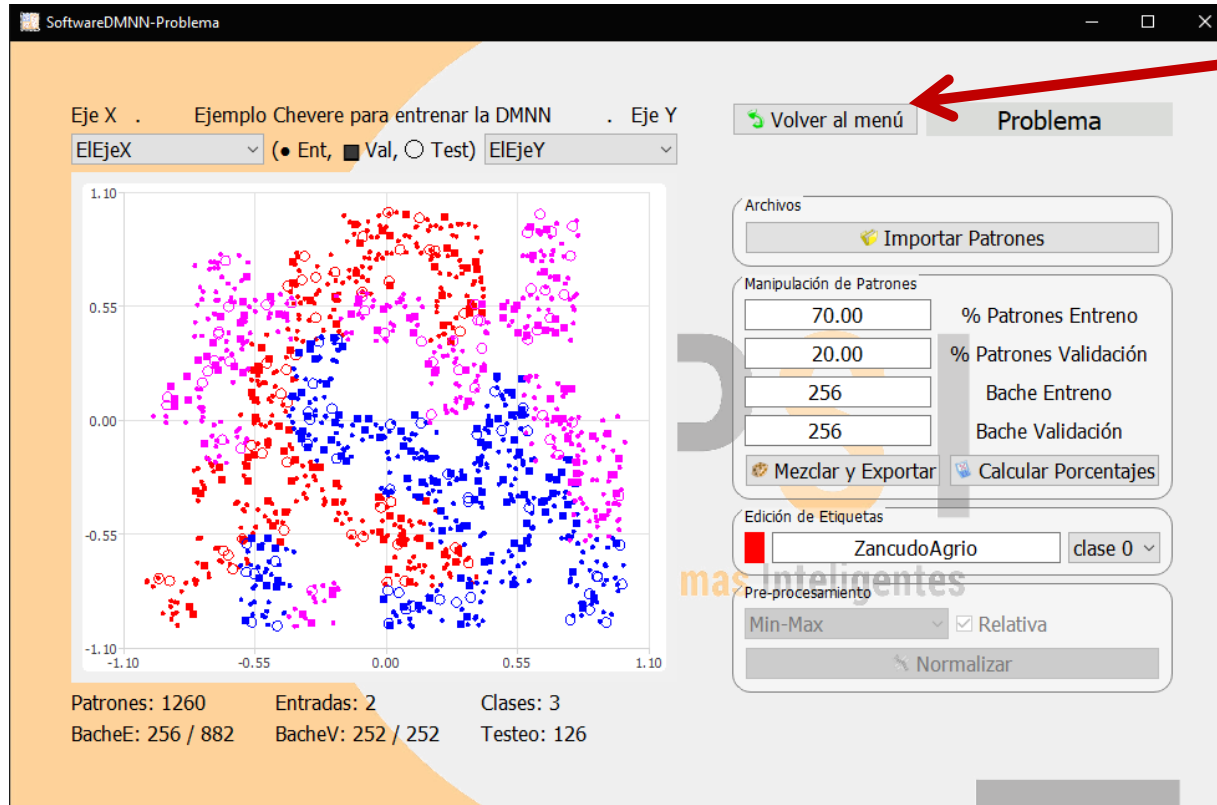
Cuenta con dos tipos de normalización: Min-Max y Z-score, se explicarán luego del tutorial



La normalización puede ser relativa, es decir independiente para cada dimensión, o absoluta, tomando todo en conjunto, se explicará luego del tutorial

Pulse "Normalizar"

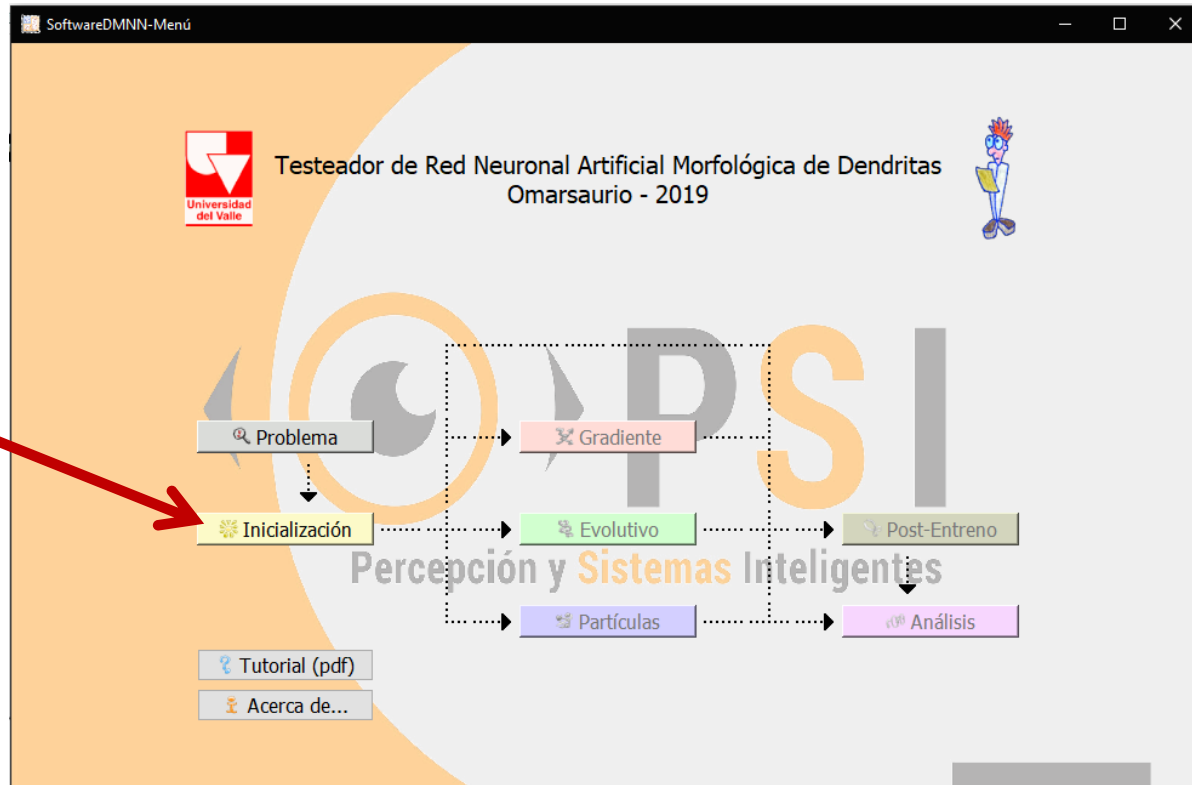
GUI - Problema



Pulse “Volver al Menú”

GUI - Menú

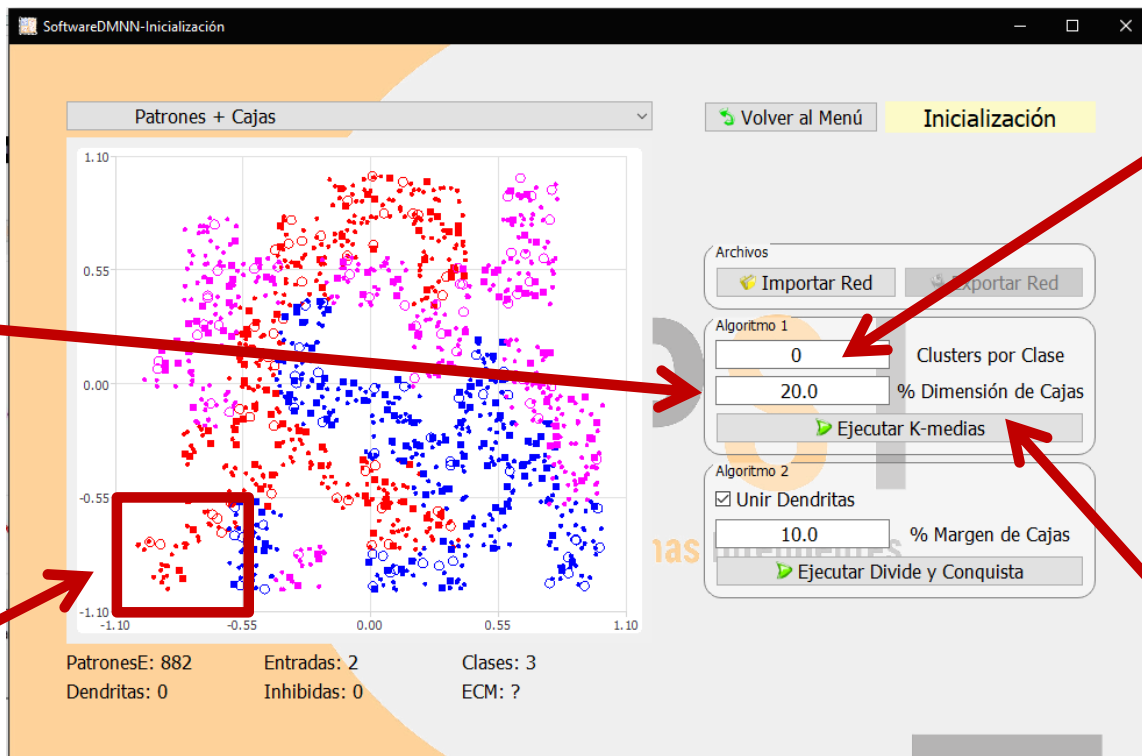
Pulse
“Iniciación”,
allí podrá crear
o importar una
red



GUI - Inicialización

El tamaño final de las hiper-cajas, el valor mínimo será limitado a 5% y el 100% referirá al 25% del tamaño total para cada dimensión

Es decir, un 100% son cajas del tamaño de la cuadrícula de la imagen

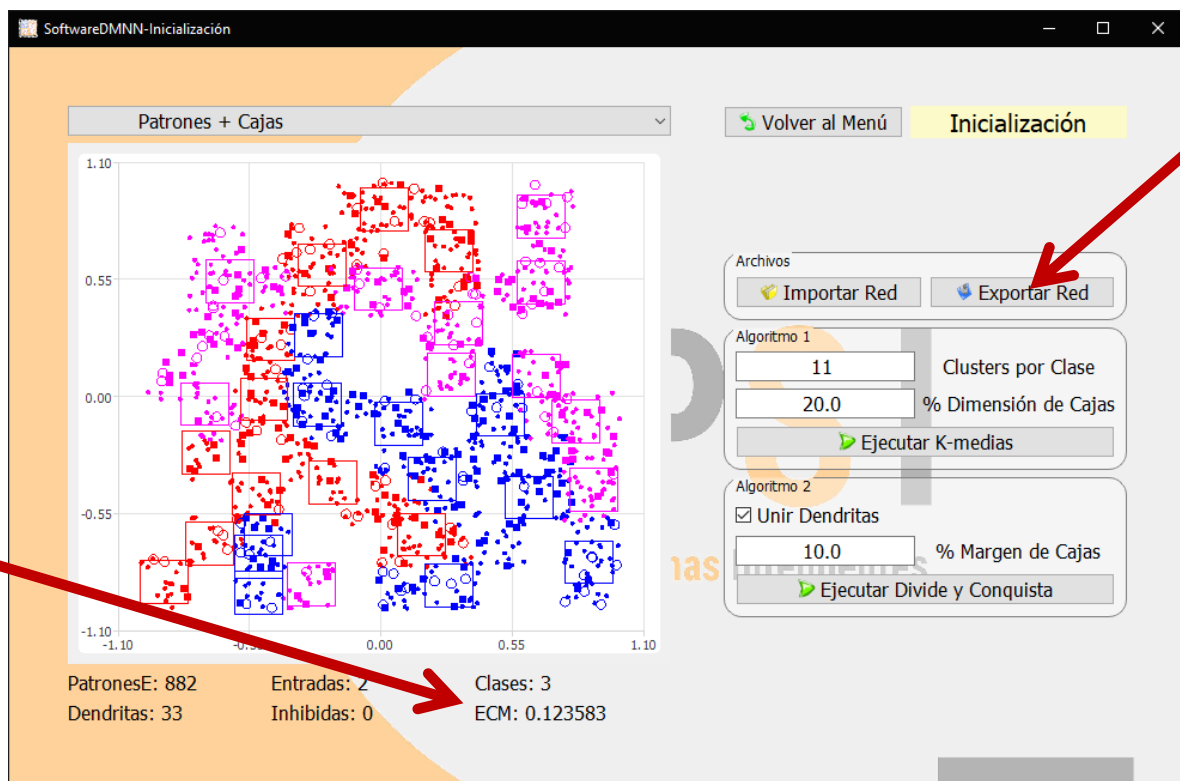


El número de clústers será la cantidad máxima de cajas posible en cada clase, puede disminuir en función de la relación de patrones por clase

Ejemplo valor 10:
C0 = 80 datos y
C1 = 40 datos,
luego
C0 = 10 cajas y
C1 = 5 cajas

Hay 2 algoritmos disponibles para crear la DMNN, el primero es K-medias.

GUI - Inicialización

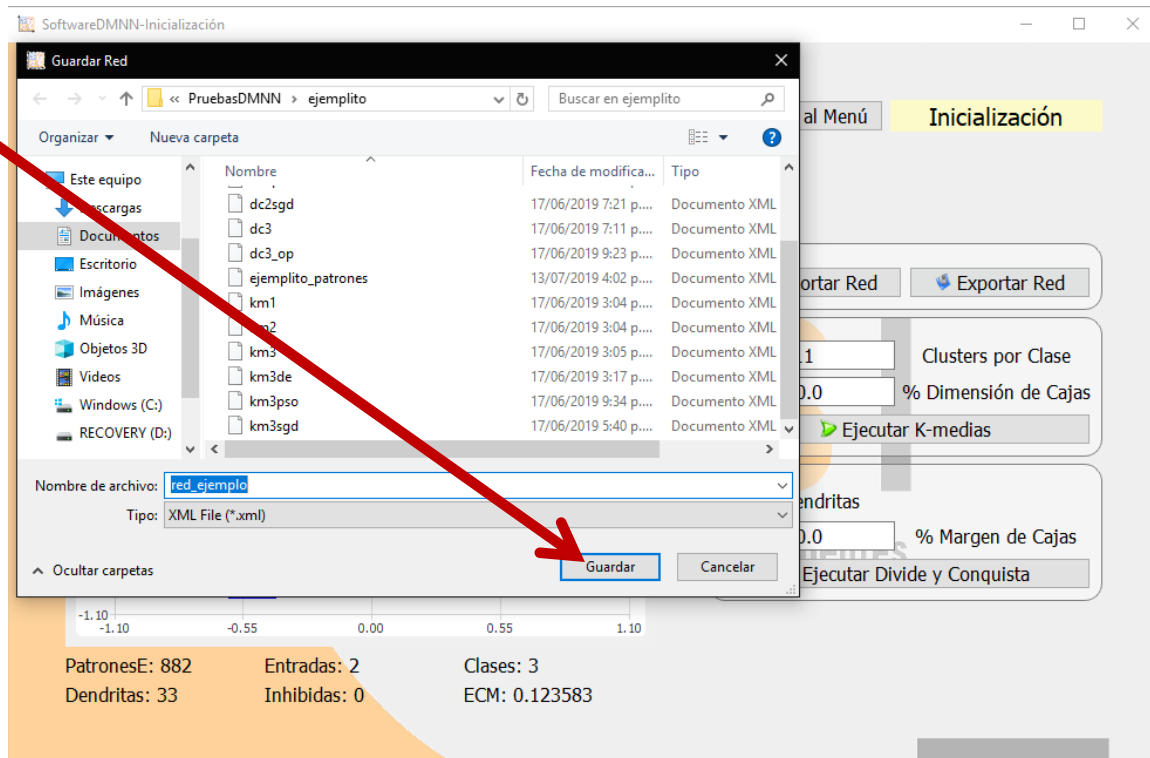


Ahora es posible exportar la red, pulse este botón para guardarla

Note el ECM (error cuadrático medio) obtenido

GUI - Inicialización

Seleccione su archivo, puede ser TXT o XML



Se recomienda usar el archivo XML por ser más robusto

Formato Red TXT

El texto "DMNN: " es obligatorio y debe ir en la primera línea del archivo, le sigue el título del problema

DMNN: TiposDeRanas

Dimension: Entradas, Clases

3,2

Pesos

4.575,4.425,6.25,6.15,10.5,9.1,4.275,4.125,6.05,5.95,7.7,6.3

DendritasPorClase

1,1

Activas

1,1

NormalizacionH

1.,1.,1.

NormalizacionL

-1.,-1.,-1.

NormalizacionN: 0.0

NombresSalidas: RanaAgridulce, RanaMaliciosa

NombresEntradas: Longitud(cm), Grosor(cm), Peso(g)

|

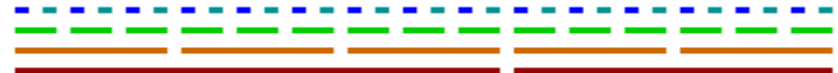
Estos son los pesos sinápticos como tal

El número de dendritas que posee cada neurona / clase

Normalización N en 0 significa sin normalización, en -1 es Z-score y >0 es Min-Max

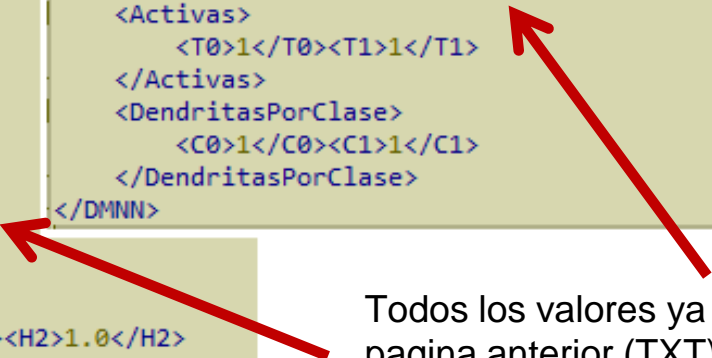
Para Z-Score: H son los promedios y L son las desviaciones estándar de cada dimensión, luego para Min-Max H son los máximos y L los mínimos respectivamente

Pesos: H,L – verde entrada – caqui dendrita – marrón neurona



Formato Red XML

```
<DMNN>
  <Titulo>TiposDeRanas</Titulo>
  <Dimension>
    <Entradas>3</Entradas>
    <Clases>2</Clases>
  </Dimension>
  <NombresSalidas>
    <A0>RanaAgridulce</A0>
    <A1>RanaMaliciosa</A1>
  </NombresSalidas>
  <NombresEntradas>
    <N0>Longitud(cm)</N0>
    <N1>Grosor(cm)</N1>
    <N2>Peso(g)</N2>
  </NombresEntradas>
  <NormalizacionH>
    <H0>1.0</H0><H1>1.0</H1><H2>1.0</H2>
  </NormalizacionH>
  <NormalizacionL>
    <L0>-1.0</L0><L1>-1.0</L1><L2>-1.0</L2>
  </NormalizacionL>
  <NormalizacionN>0.0</NormalizacionN>
  <Pesos>
    <W0>4.575</W0><W1>4.425</W1><W2>6.25</W2><W3>6.15</W3>
    <W4>10.5</W4><W5>9.1</W5><W6>4.275</W6><W7>4.125</W7>
    <W8>6.05</W8><W9>5.95</W9><W10>7.7</W10><W11>6.3</W11>
  </Pesos>
  <Activas>
    <T0>1</T0><T1>1</T1>
  </Activas>
  <DendritasPorClase>
    <C0>1</C0><C1>1</C1>
  </DendritasPorClase>
</DMNN>
```



Todos los valores ya fueron explicados en la pagina anterior (TXT), aquí cambia solo la presentación

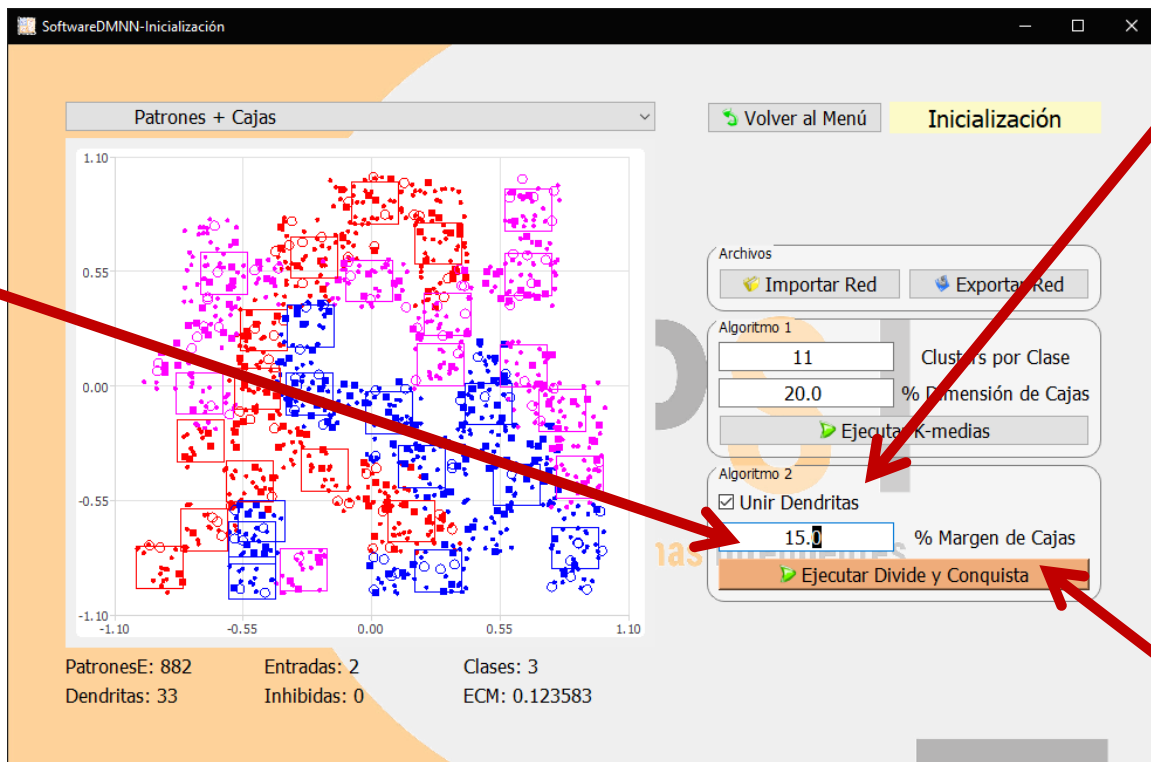
No cambie los nombres de los <elementos>

GUI - Inicialización

Se establece una margen entre las cajas, un valor de 0% fuerza a $ECM=0$, mientras mayor lo aumenta

Un valor de 100% referirá al 10% del tamaño total para cada dimensión

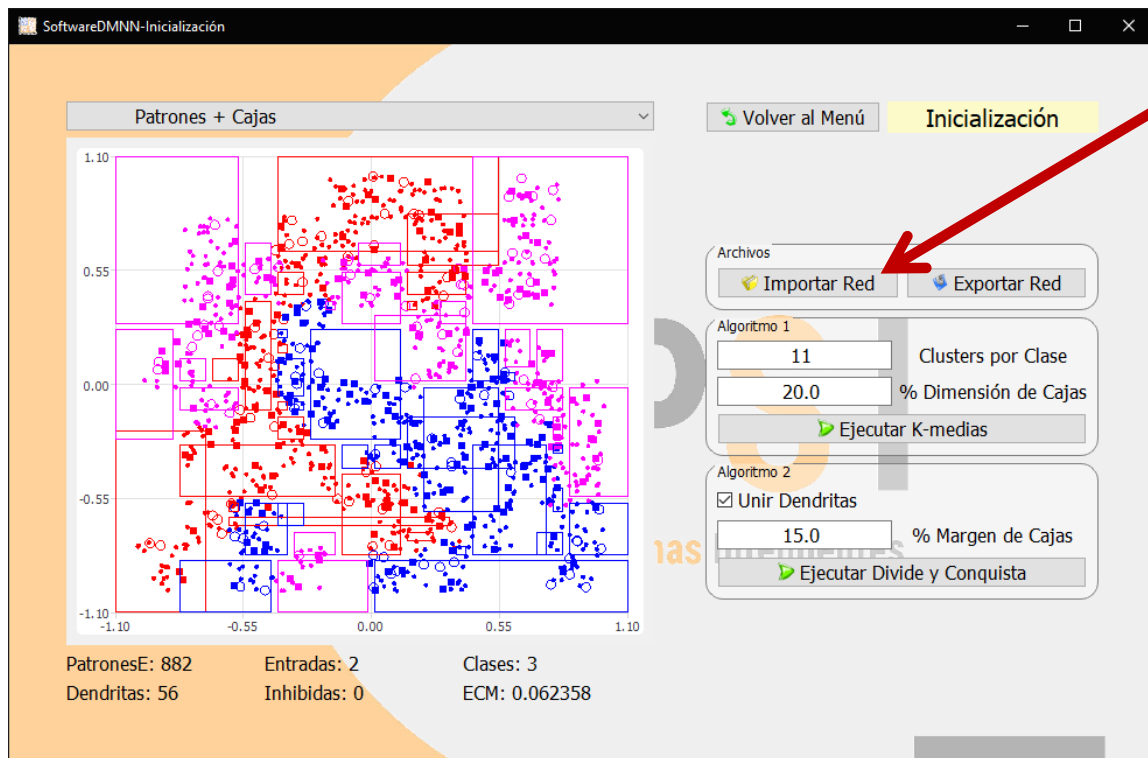
Es decir, una separación de 1/10 del tamaño de la gráfica



Unir dendritas hace un segundo ciclo que como su nombre lo indica, une dendritas que pertenezcan a la misma clase y abarquen juntas solo a datos de dicha clase (ralentiza)

El segundo algoritmo es Divide y -conquista

GUI - Inicialización



Ahora vamos a “Importar Red” para recuperar la creada mediante K-medias

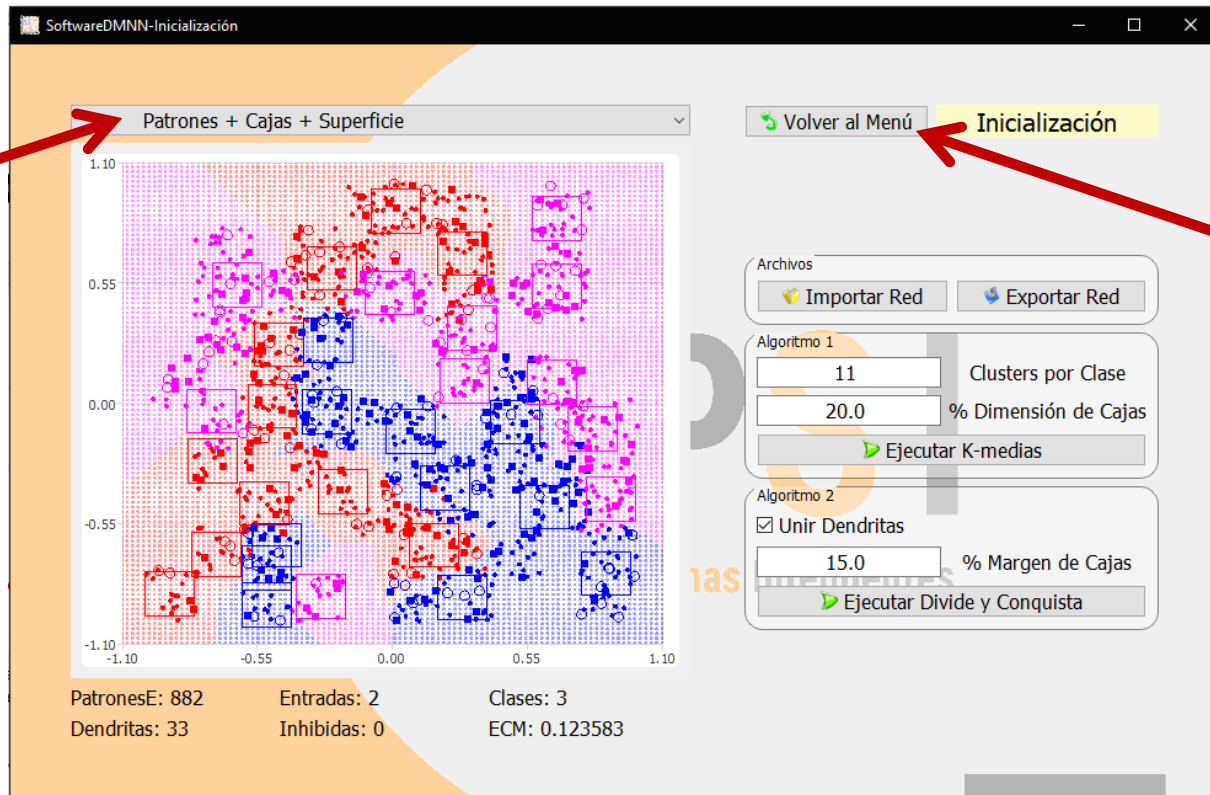
Luego solo debe buscar el archivo, seleccionarlo y pulsar abrir

GUI - Inicialización

En varias GUI's
vera que puede
graficar:

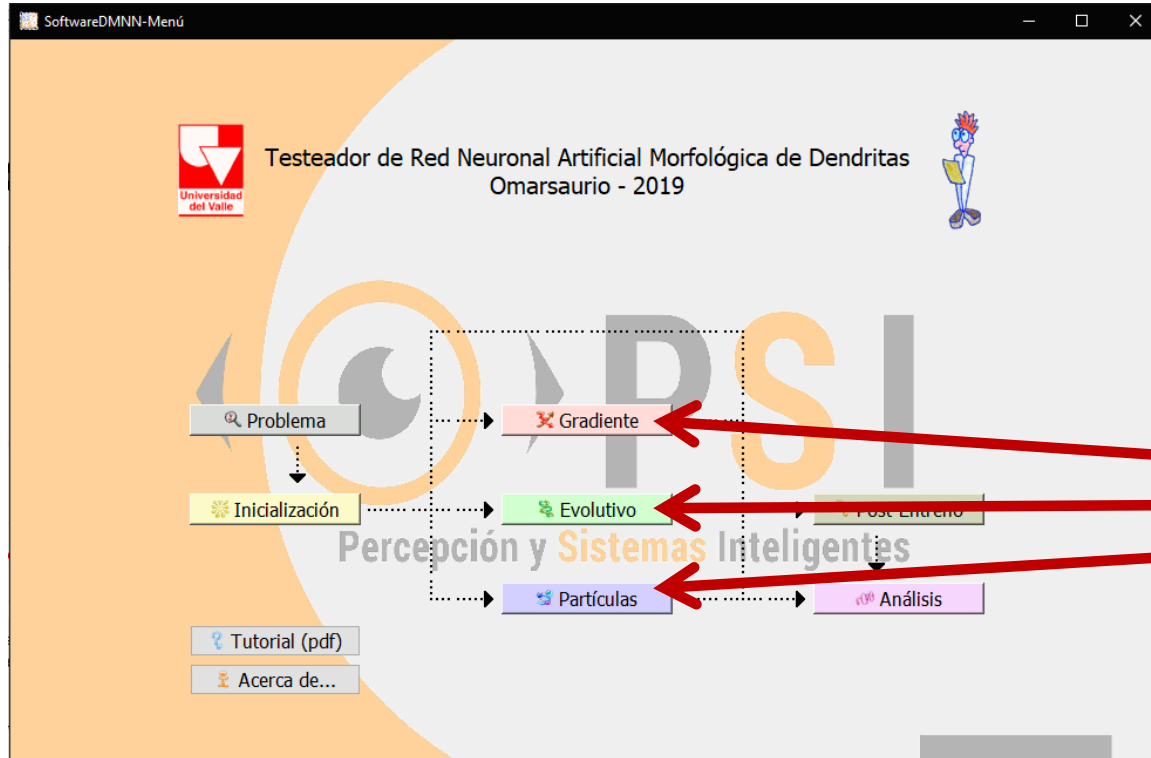
- Patrones
- Cajas
- Superficie de
decisión

Esta última toma
un poco de tiempo
y no es muy útil en
problemas de
dimensionalidad
mayor a 2, también
se desactivará al
graficar durante
entreno



Regrese el
menú
principal

GUI - Menú



Ahora están desbloqueados todos los módulos del software, hay tres encargados de entrenar la DMNN:

- Gradiente
- Evolutivo
- Partículas

Dirijámonos al primero

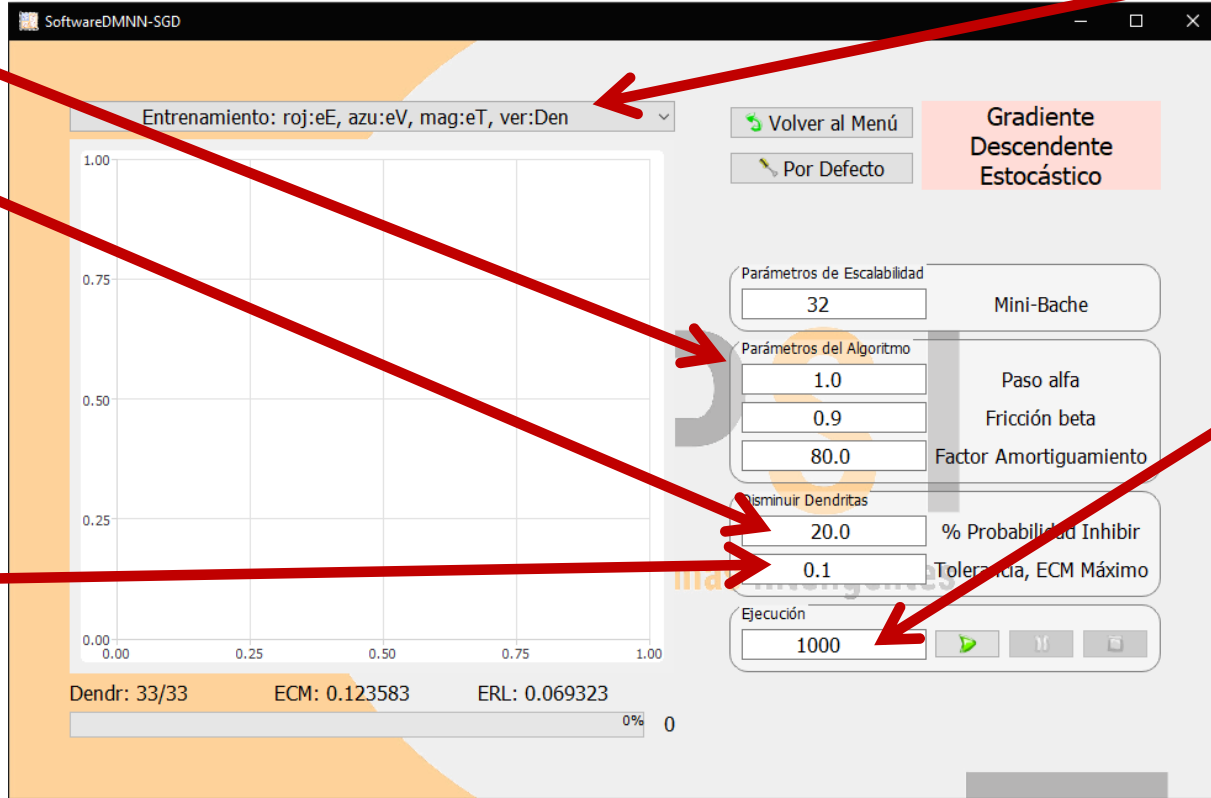
GUI - Entreno

Parámetros propios del algoritmo

Probabilidad de que en un ciclo se intente quitar una dendrita al azar, con la condición de que ECM no sobrepase la tolerancia

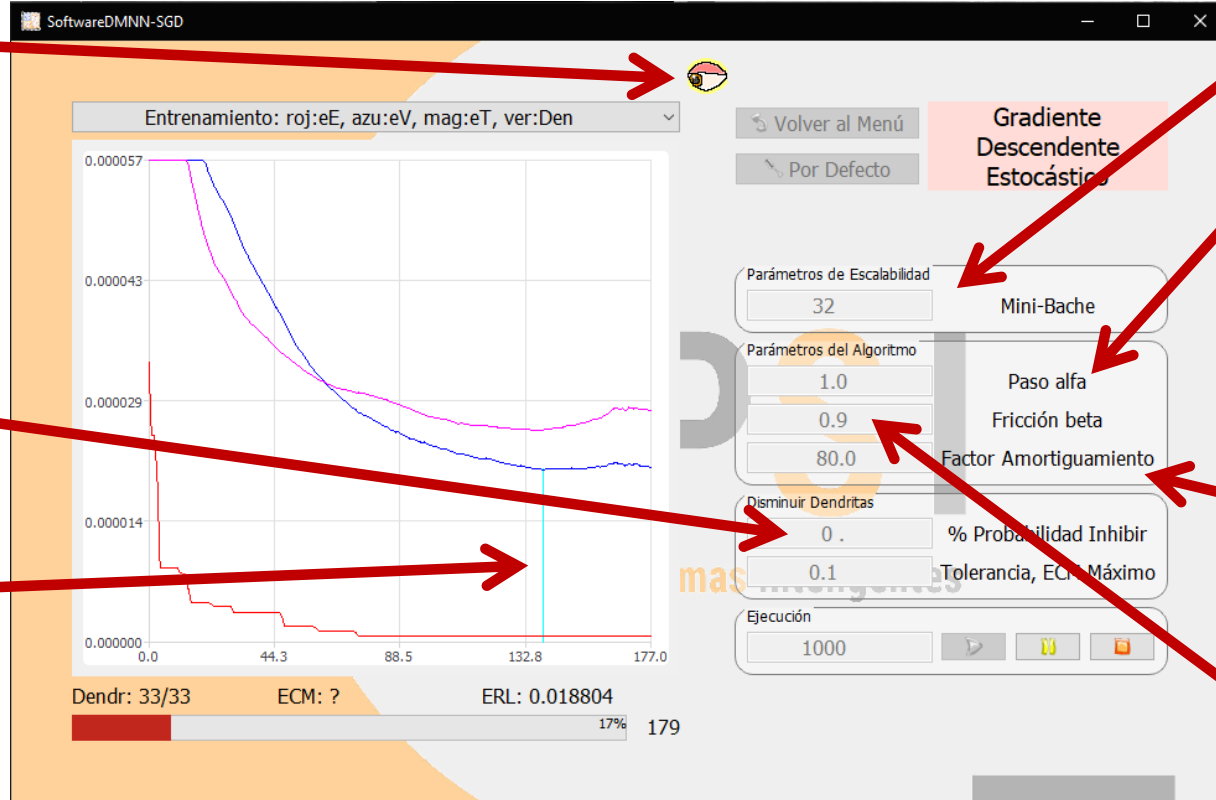
Puede visualizar las cajas en movimiento, la grafica de errores o el movimiento de los agentes

Controles de ejecución, digite el número de ciclos a iterar



GUI - SGD

Indicador de actividad



La disminución de dendritas desactivada acelera el proceso, luego puede aplicarse en post-entreno

La línea agua-marina indica el menor error de validación, red salvada

Mini-Batch es la subdivisión del batch de entreno

El paso alfa es el parámetro que define el cambio de los pesos sinápticos en cada iteración, puede disminuir con el factor de amortiguamiento

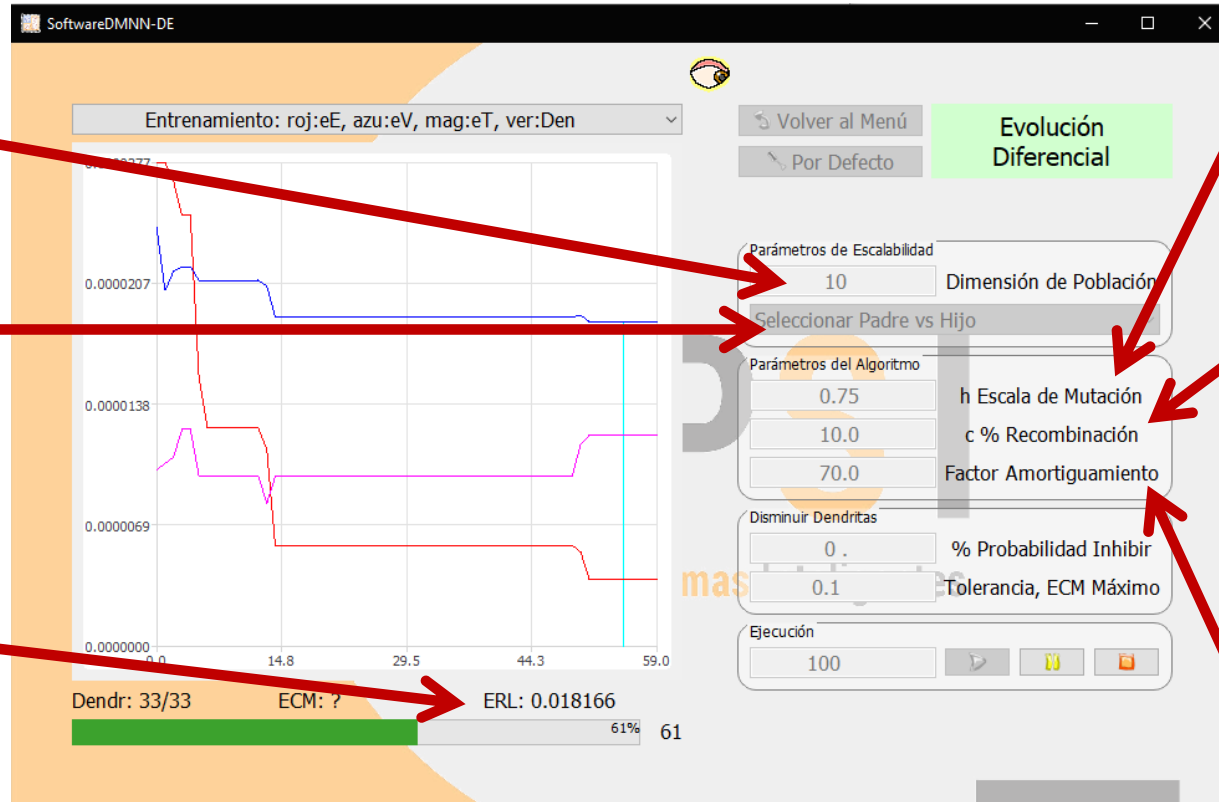
La fricción beta puede estar entre 0 y 1, inactiva o full activa

GUI - DE

Aquí se especifica el tamaño de la población

La selección padre vs hijo es recomendada dado que no converge tan rápido

El entreno se mide con el ERL (error de regresión logística), No con el ECM



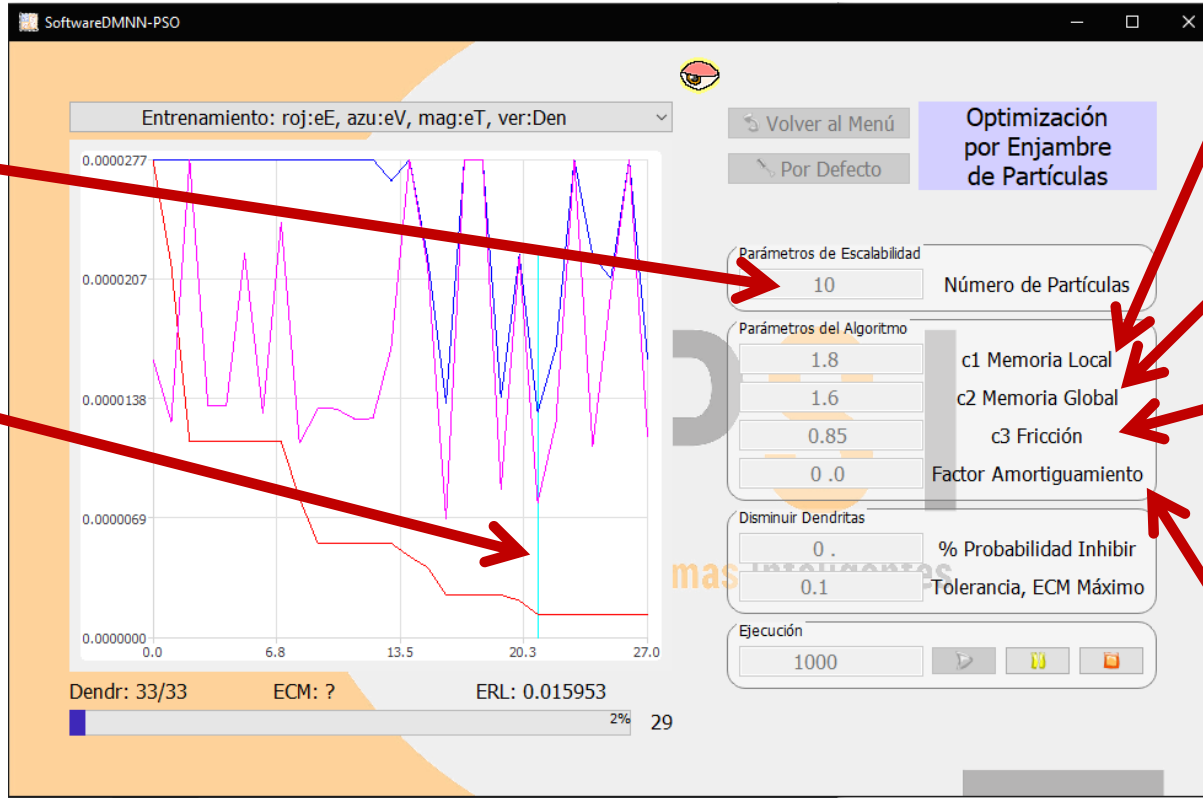
h es la fuerza de mutación, es decir el cambio en los pesos sinápticos y c el porcentaje de recombinación o similitud entre padre e hijo

h puede variar con el factor de amortiguamiento

GUI - PSO

Aquí define el número de partículas (agentes) involucrados

Note como la línea agua marina permanece en el pico menor, esa es la red que se tendrá en cuenta; hay veces el pico no es dibujado porque la gráfica se filtra cada 300 épocas



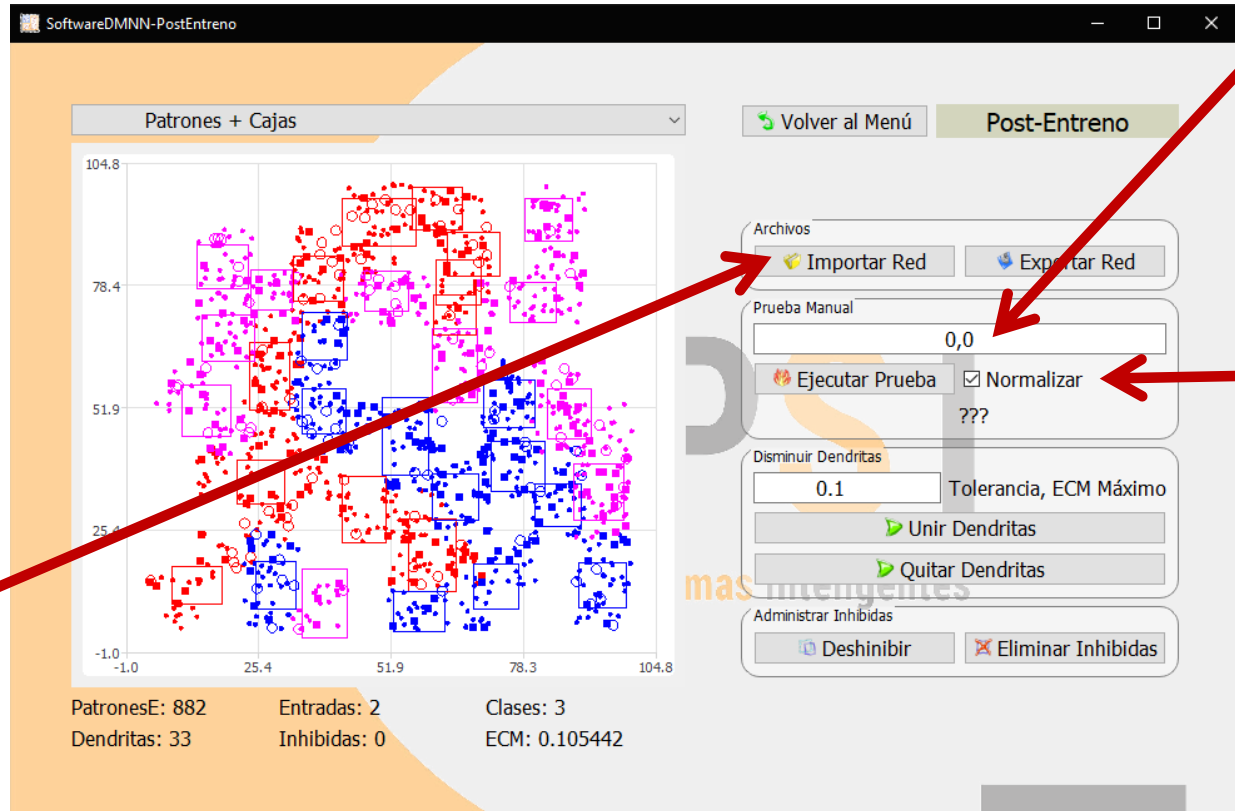
c1 es la influencia que mueve a la partícula hacia su mejor posición hallada, y c2 hacia la mejor posición global

c3 se recomienda con valores altos, cerca a 1 para no converger rápidamente; decrece con el factor de amortiguamiento

GUI - POST-ENTRENO

Luego de regresar al menú principal, entre a “Post-entreno”, aquí podrá reducir el número de dendritas (cajas) y probar la red

Las funciones de importar y exportar red son las mismas que en el GUI “Inicialización”



Digite las entradas que desea verificar y pulse “Ejecutar Prueba”

Ejemplo: 14,16

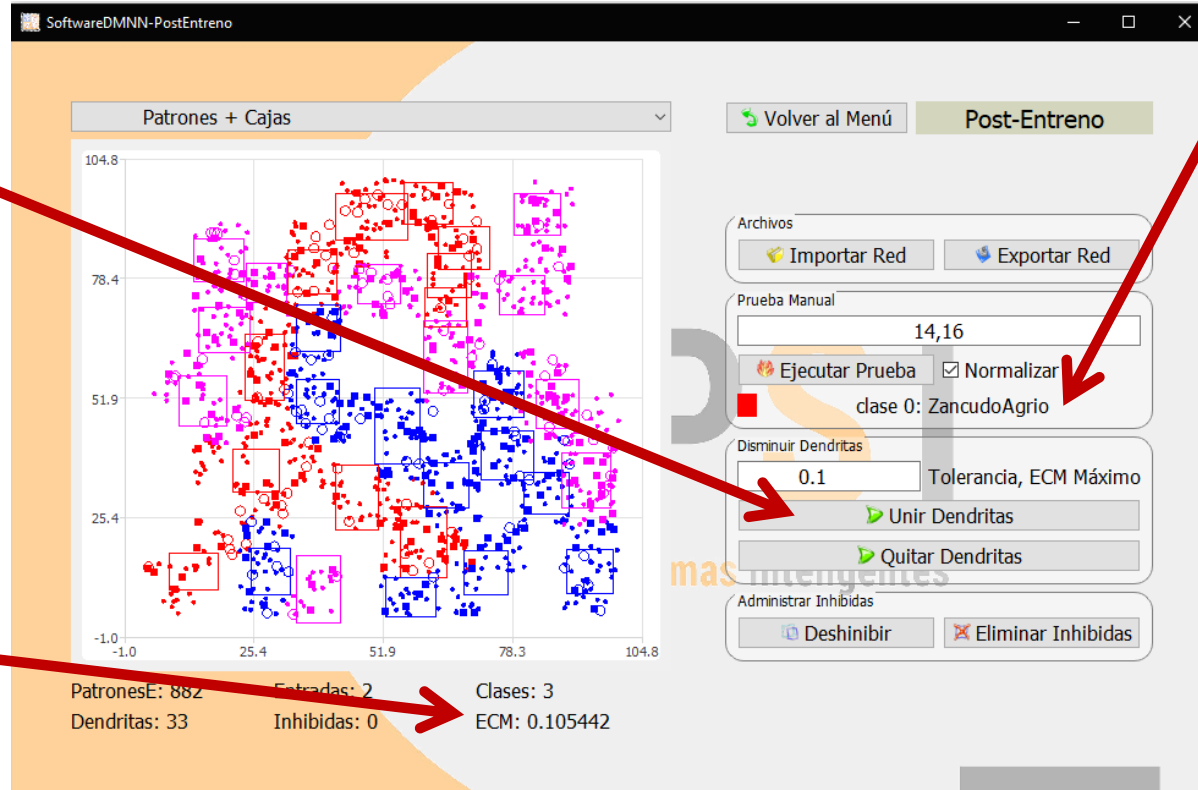
Normalizar aplicará en las entradas la misma transformación hecha al set de datos en el GUI “Problema”, si no se hizo, esta casilla será indiferente

GUI - POST-ENTRENO

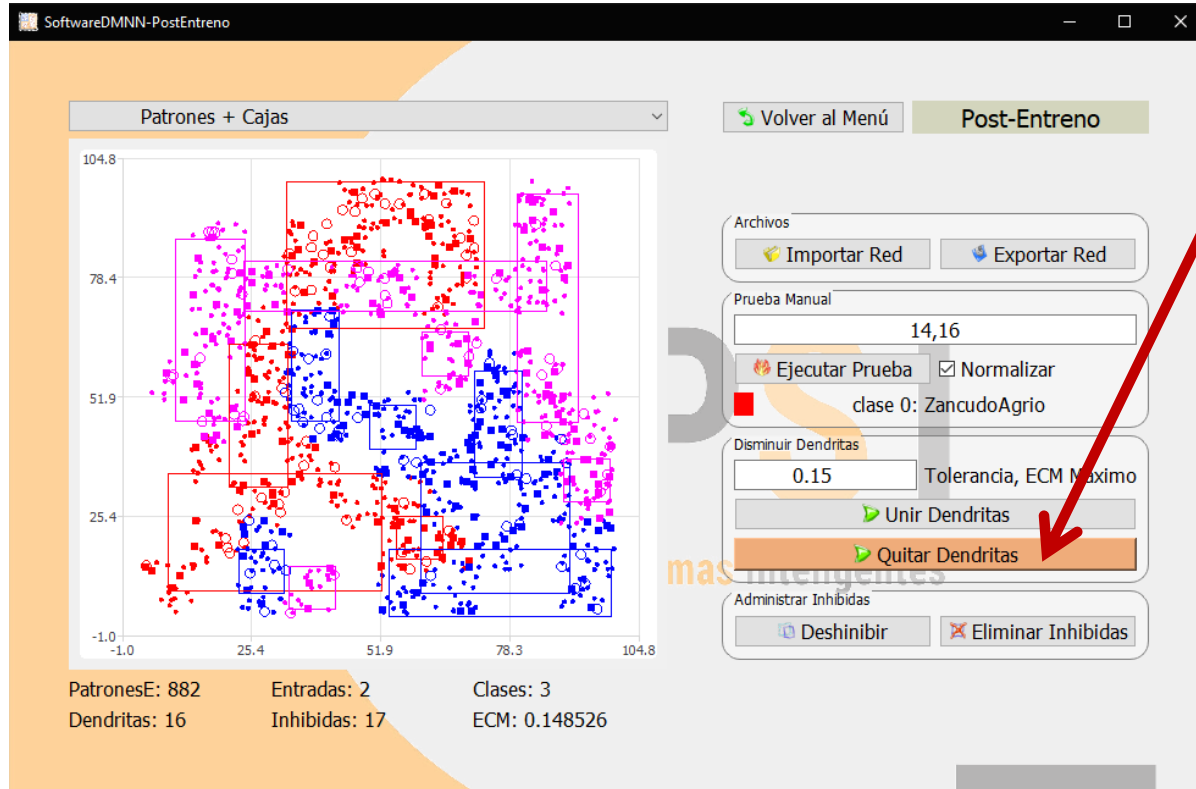
Aquí verá la salida obtenida

El primer método para reducir dendritas es tratar de unir las de la misma clase, que no aumenten el ECM mas allá de la tolerancia

Digite una tolerancia mayor al actual ECM y pulse "Unir Dendritas"



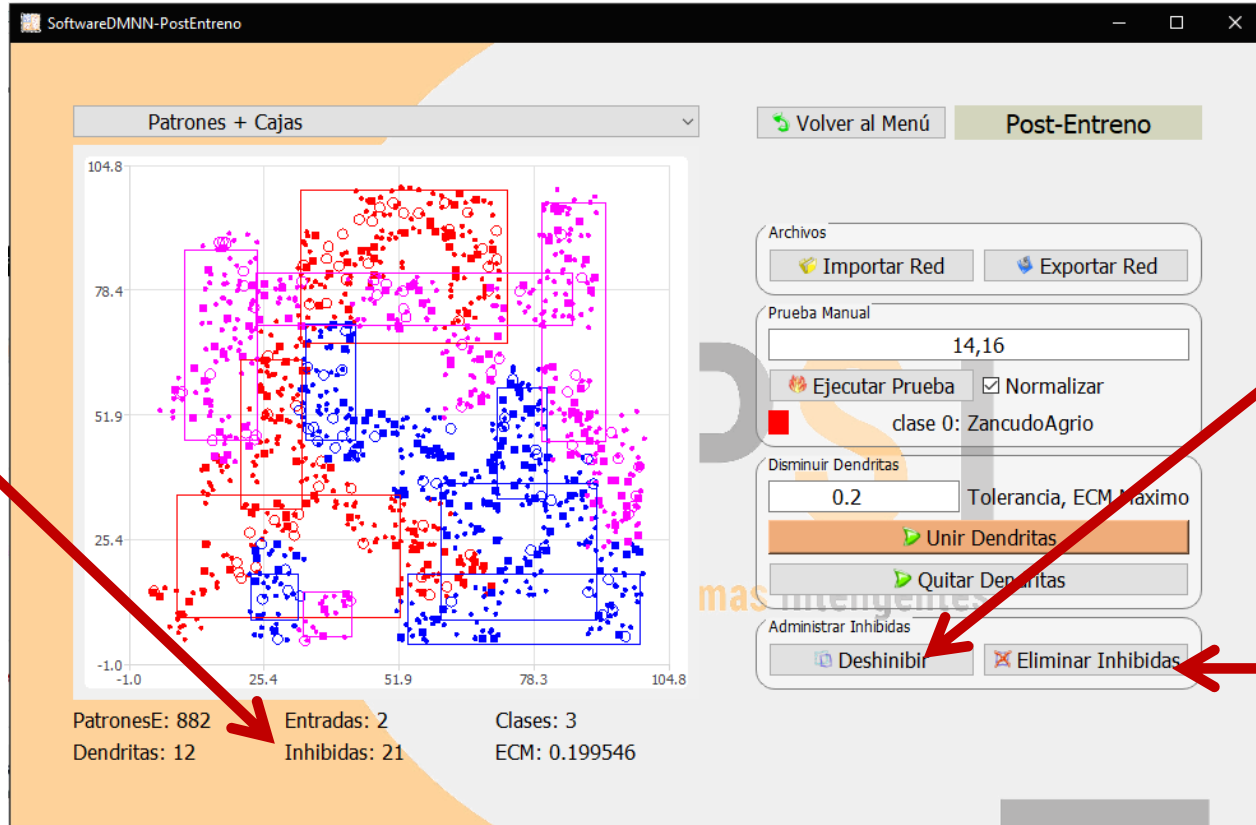
GUI - POST-ENTRENO



El segundo método es verificar si quitando cada dendrita el ECM permanece por debajo de la tolerancia, repita el paso anterior pulsando “Quitar Dendritas”

GUI - POST-ENTRENO

El software hace autoguardados al entrenar o quitar dendritas, deben aparecer archivos entorno al ejecutable

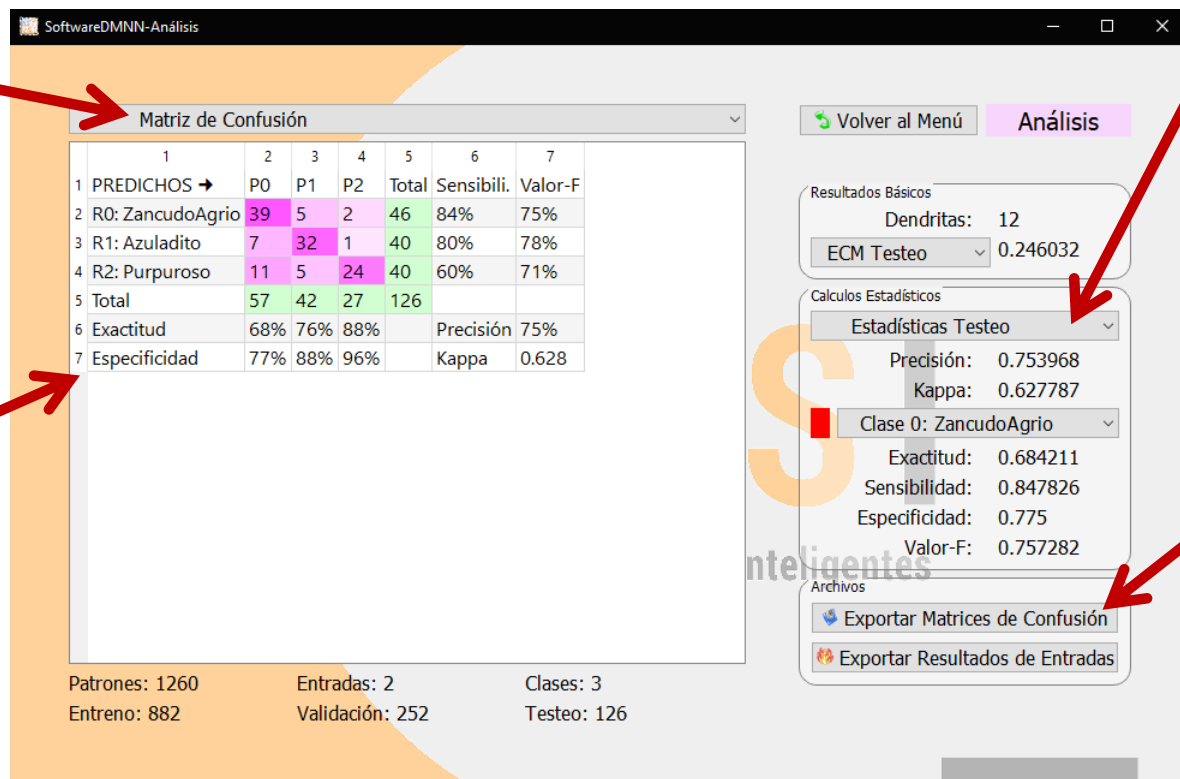


GUI - Análisis

Puede elegir si ver matrices de confusión, curva ROC o los gráficos antes vistos

La matriz de confusión posee todas las estadísticas del sistema de clasificación

P: predichos
R: reales

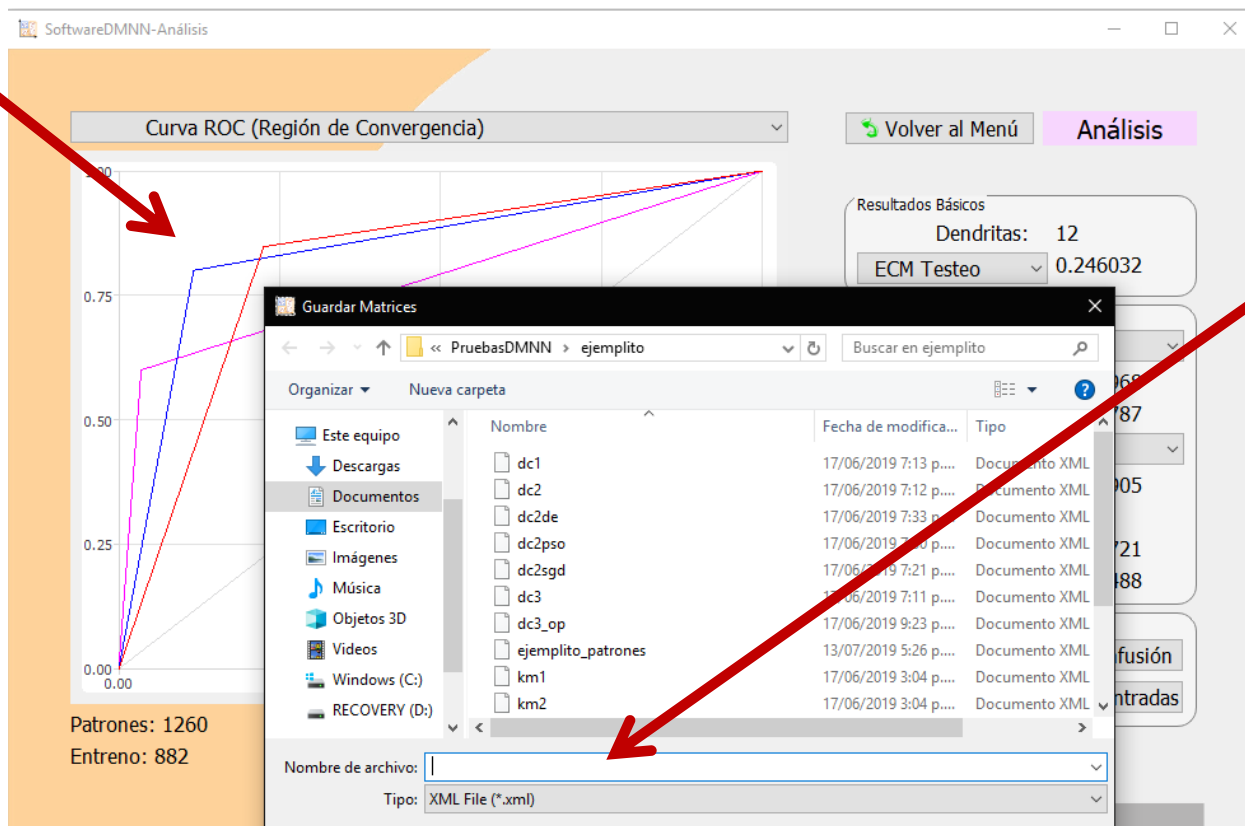


Puede seleccionar el set de datos que quiere ver en forma de matriz y estadísticas

Pulse "Exportar Matrices de Confusión", esto guardará todas las matrices en un XML, sin sus estadísticas

GUI - Análisis

Observe la ROC, entre más cerca estén las líneas al punto 0,1 mejor será el sistema clasificador



Seleccione el archivo y guarde las matrices de confusión

Formato Matrices de Confusión XML

Este XML es fácilmente legible por Excel, no cambie los nombres de sus <elementos>

Se entiende que las otras matrices (como la general) son la combinación (suma) de estas 3 básicas, y las estadísticas pueden calcularse con software externo

Entreno

Validación

Testeo

Número de dendritas

```
<dataset>Matrices de Confusion DMNM TiposDeRanas
  <record>
    <Reales>Entreno</Reales><P0 /><P1 />
  </record><record>
    <Reales>R0: RanaAgridulce</Reales>
    <P0>1</P0><P1>0</P1>
  </record><record>
    <Reales>R1: RanaMaliciosa</Reales>
    <P0>0</P0><P1>1</P1>
  </record><record>
    <Reales>Validacion</Reales><P0 /><P1 />
  </record><record>
    <Reales>R0: RanaAgridulce</Reales>
    <P0>0</P0><P1>1</P1>
  </record><record>
```

```
    <Reales>R1: RanaMaliciosa</Reales>
    <P0>0</P0><P1>0</P1>
  </record><record>
    <Reales>Testeo</Reales><P0 /><P1 />
  </record><record>
    <Reales>R0: RanaAgridulce</Reales>
    <P0>0</P0><P1>0</P1>
  </record><record>
    <Reales>R1: RanaMaliciosa</Reales>
    <P0>0</P0><P1>1</P1>
  </record><record>
    <Reales>Dendritas</Reales><P0 /><P1 />
  </record><record>
    <Reales>2</Reales><P0 /><P1 />
  </record>
</dataset>
```


GUI - Análisis

El comando aquí expuesto, es similar al de “Ejecutar Prueba” en el GUI “Post-entreno”, pero tomará a todo el set de datos como entradas

Esto es útil si desea obtener resultados masivos de nuevos datos

The screenshot displays the 'SoftwareDMNN-Análisis' application interface. On the left, a 'Matriz de Confusión' (Confusion Matrix) is shown with columns 1-7 and rows 1-7. The data is as follows:

	1	2	3	4	5	6	7
1 PREDICHOS →		P0	P1	P2	Total	Sensibili.	Valor-F
2 R0: ZancudoAgrio		353	48	19	420	84%	77%
3 R1: Azuladito		52	364	4	420	86%	81%
4 R2: Purpuroso		83	62	275	420	65%	76%
5 Total							
6 Exactitud							
7 Especificidad							

On the right, 'Resultados Básicos' (Basic Results) are displayed: Dendritas: 12, ERL Validación: 0.128056. Below this, 'Generales' (General) settings are shown with various numerical values. A 'Guardar Resultados' (Save Results) dialog box is open in the foreground, showing a file explorer view of the 'ejemplito' folder. The dialog has a 'Nombre de archivo:' field and a 'Tipo:' dropdown set to 'XML File (*.xml)'. A red arrow points from the 'Exportar Resultados de Entradas' button in the background to the 'Nombre de archivo:' field in the dialog.

Solo debe ordenar los datos en un TXT como si fuesen patrones, importarlos, importar la red nuevamente y venir a este GUI para obtener resultados

Pulse el botón “Exportar Resultados de Entradas” y guarde el archivo XML o TXT

Formato Resultados TXT

Solo posee dos columnas, el valor deseado especificado en lo que para el software sería el set de datos (patrones)

Pero si usted está usando datos sin salida, puede importar, como se dijo antes un set con al menos una salida para cada clase, si deja todas las salidas en 0 habrá error de importación

```
Resultados DMNN: TiposDeRanas
Deseado, Obtenido
0,0
1,1
0,1
1,1
```

La otra columna será el valor que la red neuronal calculó

```
Patrones: TiposDeRanas
Salidas: RanaAgridulce, RanaMaliciosa
Entradas: Longitud(cm), Grosor(cm), Peso(g)
4.5, 6.2, 9.8, 0
4.7, 5.8, 6.9, 1
5.1, 5.9, 7.7, 0
4.2, 6.0, 7.0, 1
```

Al menos un 1 en salidas para este ejemplo de 2 clases

Formato Resultados XML

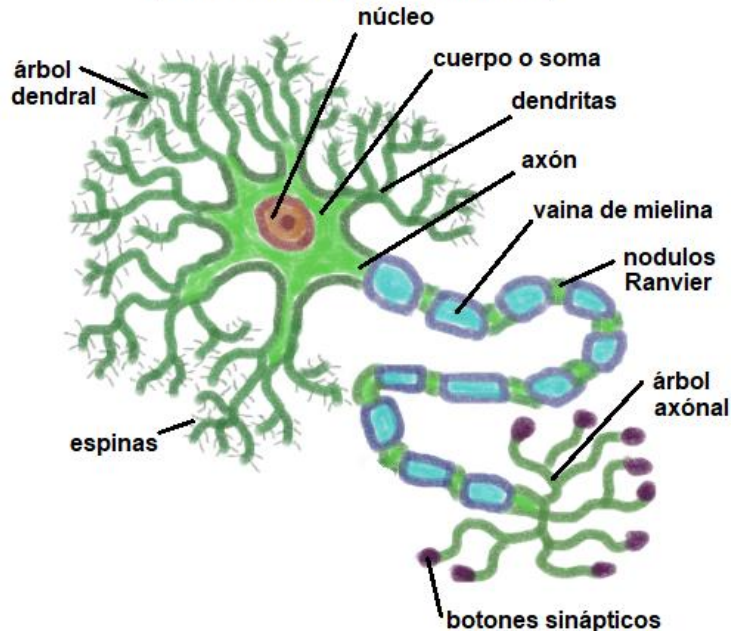
Este formato también fácilmente legible por Excel posee las mismas características del TXT

```
<dataset>Resultados DMNN: TiposDeRanas
  <record>
    <Deseado>0</Deseado><Obtenido>0</Obtenido>
  </record>
  <record>
    <Deseado>1</Deseado><Obtenido>1</Obtenido>
  </record>
  <record>
    <Deseado>0</Deseado><Obtenido>1</Obtenido>
  </record>
  <record>
    <Deseado>1</Deseado><Obtenido>1</Obtenido>
  </record>
</dataset>
```

**Enhorabuena, has
completado el
Tutorial**

Información Básica Sobre la Red Neuronal Artificial Morfológica de Dendritas y los Algoritmos Asociados

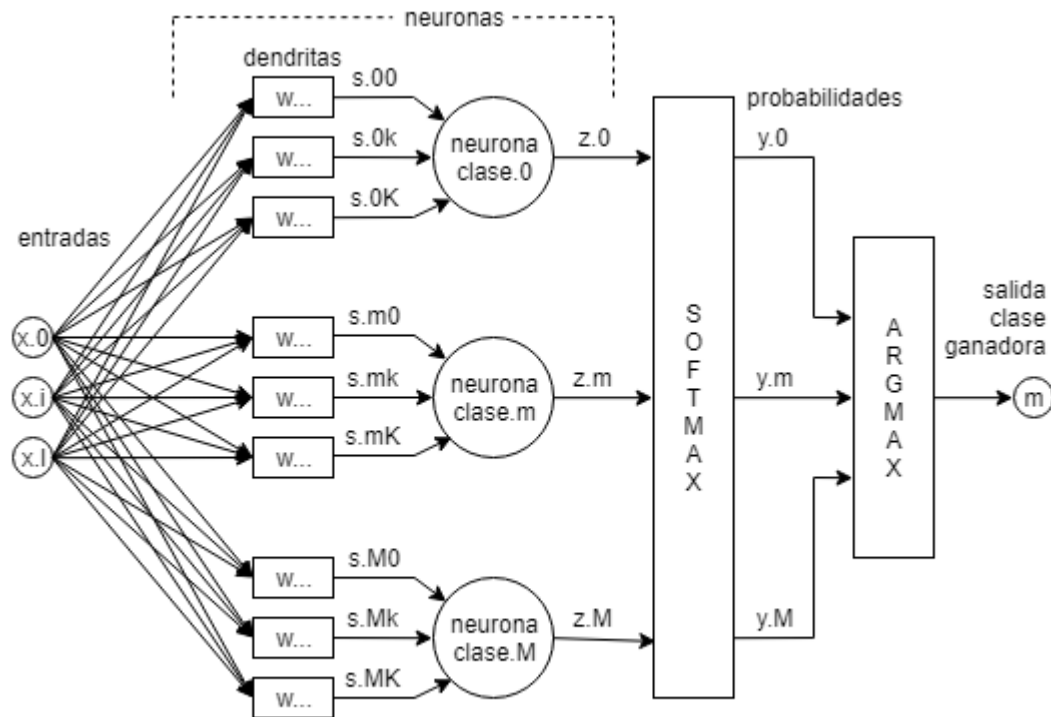
Estructura de la Neurona Biológica



La mayoría de conexiones sinápticas de la neurona biológica se producen en el árbol dendrítico de la célula, allí es donde se procesa la mayor cantidad de información, se ha propuesto que son las dendritas las unidades informáticas elementales del cerebro.

Este trabajo se fundamenta en los estudios de:
Erick Zamora, Fernando Arce, Gerhard X. Ritter, Humberto Sossa, James Kennedy, Jennifer L. Davidson, Yonggwan Won, Trelea, R. Storn, Kenneth V. Price, Laurentiu Lancu, Gonzalo Urcid.

Estructura DMNN



Red Neuronal Artificial Morfológica de Dendritas (DMNN)

Cada neurona representa a una clase, y a su entrada tiene las dendritas, las cuales se conectan totalmente a las entradas de la red (no es necesaria la capa Softmax).

Las dendritas pueden crecer o disminuir en cantidad con ciertos algoritmos, no tienen función de activación y pueden variar en número de una neurona a otra.

Está red no requiere de capas ocultas, solamente se vale del aumento de dendritas, de allí su potencial y menor complejidad.

Estructura DMNN

$$s_{mk} = \bigvee_{i=0}^I \bigvee_{l=0:L}^{1:H} (-1)^l \cdot (x_i + w_{mki}^l)$$

$$z_m = \bigwedge_{k=0}^{K_m} s_{mk}$$

$$y = \operatorname{argmax}(z_m)$$

I = número de entradas.

M = número de neuronas.

$\ell = 0-1$, $\ell := L-H$ como etiqueta.

L = low (bajo), **H** = high (alto).

X = valores de las entradas.

W = valores de pesos sinápticos.

K = valores de número de dendritas.

S = salidas de las dendritas.

Z = salidas de las neuronas.

y = clase / neurona ganadora.

En las ecuaciones de la izquierda No se tiene en cuenta la capa Softmax, respecto a la página anterior “y.m” sería el mismo “z.m” y la salida “m” aquí es “y”; se ha dividido todo en 3 partes para su comprensión:

Primero que todo, las dendritas utilizan operaciones morfológicas (suma y mínimo \vee) para operar a todas las entradas “x” por sus correspondientes pesos sinápticos “w” (alto H y bajo L); se puede ver como si cada entrada estuviese acotada por dos extremos, siendo positiva entre ellos y negativa al exterior.

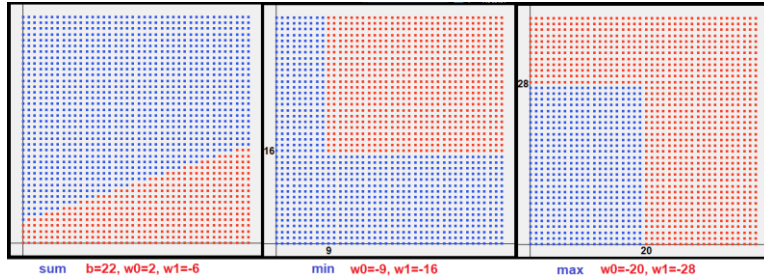
Luego la neurona escoge “z” a la mayor (\wedge) salida “s” de sus dendritas.

Finalmente Argmax escoge a la neurona “m” con mayor salida “z”.

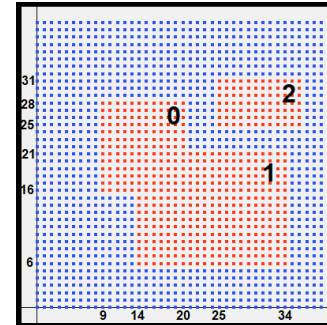
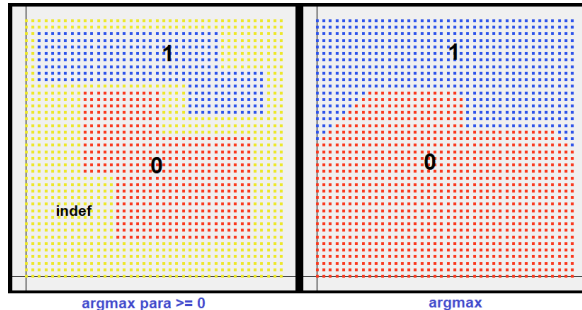
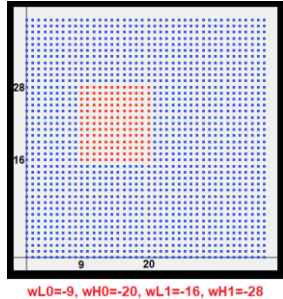
rojo (+), azul (-)

Estructura DMNN

1- A la izquierda, para **2D** tenemos la división lineal del **perceptrón**, luego dos imágenes referentes a las operaciones morfológicas, usando **min** y **max** respectivamente, ello parte cada dimensión en dos regiones con un desfase respecto a 0.



2- La **dendrita** utiliza dos operaciones morfológicas creando una caja, hipercaja para mayor dimensión.



3- Una **neurona** tiene varias dendritas (aquí 3), con lo que crea superficies de decisión complejas.

4- Varias neuronas (aquí 2: **roja** y **azul**), encierran cada una a su superficie positiva, en la imagen de la derecha se puede notar la superficie con diagonales, dado que allí compiten las dos clases mientras en los otros ejemplos se hace el corte como mayor que 0.

Estructura DMNN

```
1  import numpy as np
2  def EjecutarRed(entradas, pesW, numK):
3      X = entradas
4      while X.size < pesW.size / 2:
5          X = np.hstack((X, entradas))
6      W = pesW.copy().reshape(-1, 2)
7      WH = W[:, 0] - X
8      WL = X - W[:, 1]
9      Wmki = np.minimum(WH, WL)
10     Wmki = Wmki.reshape(-1, entradas.size)
11     Smk = Wmki.min(axis=1)
12     Zm = np.zeros(numK.size)
13     n = 0
14     for m in range(Zm.size):
15         Zm[m] = Smk[n:(n + numK[m])].max()
16         n += numK[m]
17     y = np.argmax(Zm)
18     return y
```

Este es el código en Python que hace la ejecución de la DMNN.

“**entradas**” es un vector de flotantes con los valores que se desea ingresar a la red.

“**pesW**” es un vector de flotantes con toda la cadena de pesos sinápticos.

“**numK**” es un vector de enteros que guarda el número de dendritas que posee cada neurona.

El software exporta estos dos últimos datos en formato **TXT** como ya se explicó; cabe aclarar que este código No corre la capa Softmax ni la inhibición de dendritas, por lo que supone todas las dendritas presentes como activas.

Partición de Patrones

Matriz

#, #, #, 0
#, #, #, 0
#, #, #, 0
#, #, #, 0
#, #, #, 1
#, #, #, 1
#, #, #, 1
#, #, #, 1
#, #, #, 1
#, #, #, 2
#, #, #, 2
#, #, #, 2

Mezcla:

Como los sets de datos tienden a estar con sus clases en orden, es necesario hacer aleatorización.

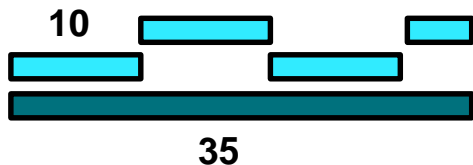
Matriz

#, #, #, 1
#, #, #, 1
#, #, #, 2
#, #, #, 0
#, #, #, 1
#, #, #, 2
#, #, #, 0
#, #, #, 2
#, #, #, 0
#, #, #, 1
#, #, #, 1
#, #, #, 0

Entreno: set usado para modificar los pesos de la red durante el entreno.

Validación: set para evitar sobre-entrenamiento, la mejor solución corresponde al mínimo error de validación.

Testeo: set no involucrado en la búsqueda de la solución, prueba final.



Batches: En el ejemplo, un set de entreno (oscuro) de 35 datos, es dividido en batches (claros) de 10 datos (uno queda de 5); esto favorece la velocidad de entreno y agrega ruido que puede sacar de mínimos locales.

Pre-Procesamiento de Patrones

Z-Score: utiliza el promedio y desviación estándar extraídos del set para redimensionar los datos, “v” hace referencia a un valor puntual.

$$v = \frac{v - prom}{desvstd}$$

Min-Max: cambia el rango de los datos, donde min y max son extraídos del set; nmin y nmax son los valores deseados como rango, la ecuación de arriba limita a -1,1.

$$v = 2 \cdot \frac{v - min}{max - min} - 1$$

$$v = \frac{v - min}{max - min} \cdot (nmax - nmin) + nmin$$

```
Patrones: TiposDeRanas
Salidas: RanaAgridulce, RanaMaliciosa
Entradas: Longitud(cm), Grosor(cm), Peso(g)
4.5, 6.2, 9.8, 0
4.7, 5.8, 6.9, 1
5.1, 5.9, 7.7, 0
4.2, 6.0, 7.0, 1
```



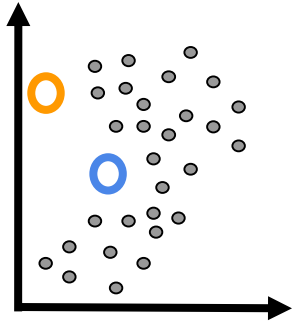
Relativo: toma cada dimensión por separado para sacar prom, desvstd, min o max.

```
Patrones: TiposDeRanas
Salidas: RanaAgridulce, RanaMaliciosa
Entradas: Longitud(cm), Grosor(cm), Peso(g)
4.5, 6.2, 9.8, 0
4.7, 5.8, 6.9, 1
5.1, 5.9, 7.7, 0
4.2, 6.0, 7.0, 1
```

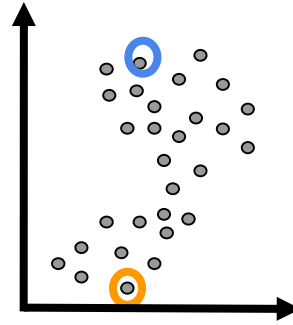


Absoluto: toma todo el set para sacar prom, desvstd, min o max.

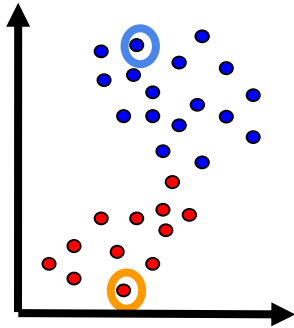
Algoritmo K-medias



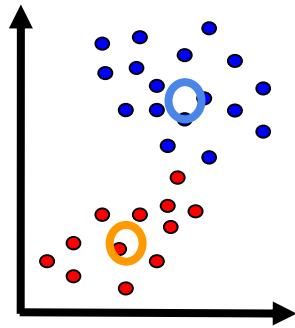
1- Este algoritmo crea clústers de datos (agrupamiento) según sus hiper-distancias, inicialmente se crea un número finito de centroides (puntos) ubicados al azar.



2- Opcionalmente se reubican los centroides para obtener un mejor esparcimiento inicial.



(a)



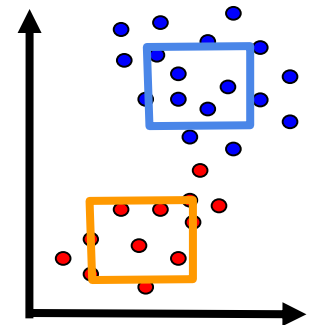
(b)

3- Se hace un proceso iterativo que consta de dos partes (y siempre converge):

a- Se asocian los patrones al centroide más cercano.

b- Se mueven los centroides al lugar promedio dado por sus patrones asociados.

4- Se crean las hipercajas en cada centroide.



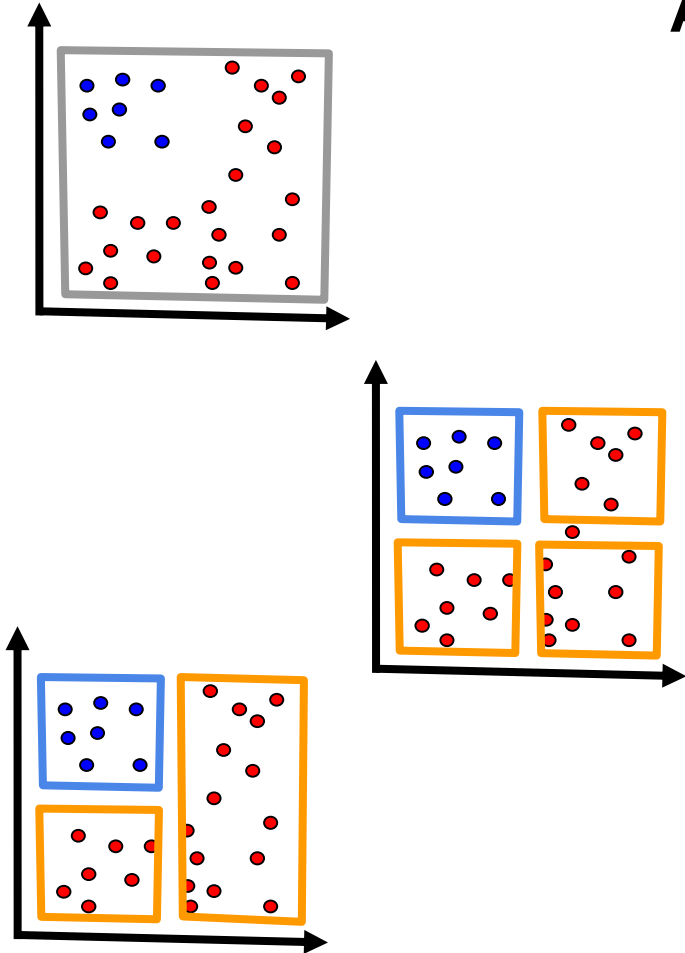
Algoritmo D&C

Este procedimiento crea en inicio una hipercaja que abarca a todos los patrones más un **10%** de tamaño, luego comienzan las iteraciones finitas en donde:

Verifica que una hipercaja encierre sólo a una clase, de ser así se asocia la caja a esa clase y no se opera más, pero de lo contrario, es dividida en dos por cada dimensión; total: 2^i cajas (i : dimensión del problema).

Esta división lleva un parámetro “**margen**” que genera un pequeño espacio entre las cajas, lo que acelera la convergencia, puesto que con $\text{margen} = 0$ el error de entreno es **0**, dando un máximo de cajas.

Luego de este ciclo finito se hace otro ciclo finito de unión de hipercajas, donde para cada una se verifica si puede unirse a otras, formando una única caja mayor con patrones de una misma clase dentro.



Algoritmo SGD

El gradiente descendente es el algoritmo de entreno más usado en redes neuronales, garantiza que el error disminuya hasta un mínimo local; para problemas de clasificación es común usar capa **Softmax** (ecuación izquierda), para derivar la salida seleccionada.

Luego la red DMNN no es derivable por sus funciones **max** y **min**, así que la ecuación del medio hace una aproximación, propagando el error de **regresión logística** hacia atrás; el “**yd**” es la salida deseada, luego “**w**” es el peso sináptico de la neurona “**m**”, dendrita “**k**”, entrada “**i**” del extremo “**H**” o “**L**” respectivamente. La aproximación lo que hace es seguir hacia atrás a la cadena de selecciones **max** y **min** para averiguar cual es este peso responsable por el resultado dado.

Finalmente tras hallar esto en cada iteración se procede a modificar el peso con la ecuación de la derecha (**GD**), donde el peso “**w**” varía con un parámetro **Alfa** (α) o paso que puede cambiar durante el entreno, y **Beta** (β) que es la fricción de la inercia “**u**”, el “**w**” de la derivada es el mismo modificado. Se llama estocástico (**SGD**) si se toman muestras al azar de todo el set de datos.

$$y_m = e^{z_m} / \sum_{n=0}^M e^{z_n}$$

$$\frac{\partial \varepsilon}{\partial w_{mki}^l} = \frac{1 - y_d}{\ln(10)} \cdot \begin{cases} 1, & l = L \\ -1, & l = H \end{cases}$$

$$u = \beta \cdot u + \frac{\partial \varepsilon}{\partial w}$$

$$w = w - \alpha \cdot u$$

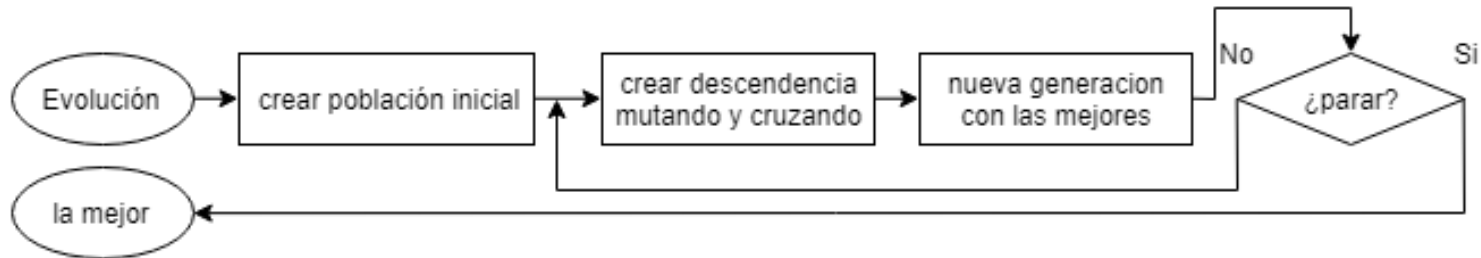
Algoritmo DE

Este algoritmo de optimización crea varias redes “n”, cada una será un agente llamado individuo, luego iterativamente reproducirá a los individuos, los hijos serán comparados con sus respectivos padres y sobrevivirá el que tenga un menor error, para mantener la población de tamaño “n”.

En la ecuación se ve el proceso de reproducción del individuo enésimo “sub-n” generando al “u”, para ello se eligen al azar otros 3 individuos “x0”, “x1”, “x2”; el subíndice “i” refiere a cada peso sináptico operado, las constantes “h” y “c” son para la fuerza de la mutación y el porcentaje de recombinación respectivamente (“c” generalmente bajo).

Existen en estos algoritmos evolutivos diferentes formas de reproducir y seleccionar individuos.

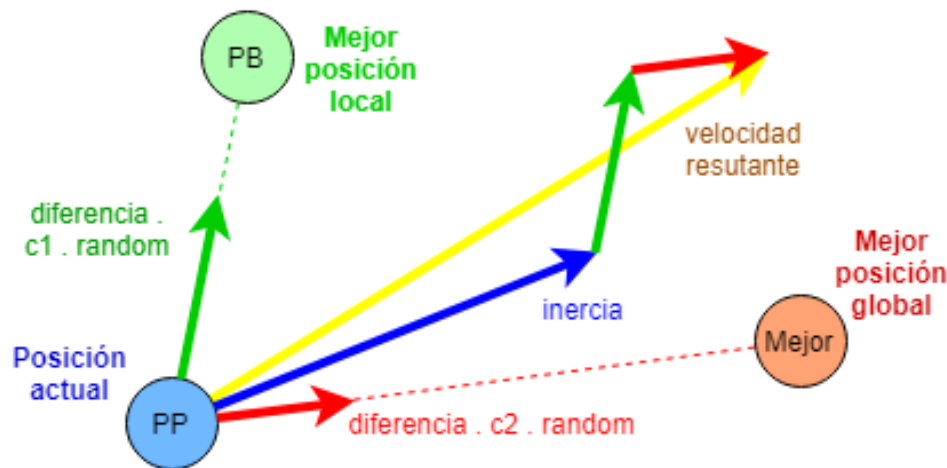
$$u_i = \begin{cases} Prob_{x_{0i}} + h \cdot (Prob_{x_{1i}} - Prob_{x_{2i}}), & rand < c \\ Prob_{ni}, & else \end{cases}$$



Algoritmo PSO

$$PV_n = c3.PV_n + c1.rand.(PB_n - PP_n) + c2.rand.(Mejor - PP_n)$$

$$PP_n = PP_n + PV_n$$



Este algoritmo de optimización crea varias redes “n”, cada una será un agente llamado partícula, el cual posee una posición “PP” que es el valor de sus pesos sinápticos, posee una posición “PB” que es la que tuvo el error más bajo para esa partícula, y posee una velocidad “PV” para cada peso sináptico.

La ecuación muestra la actualización del movimiento que rige a la partícula, donde “C1” y “C2” favorecen exploración y explotación respectivamente, “C3” es la fricción de la inercia.

“Mejor” es la posición “PB” de la partícula que alcanzó el mínimo error.

Factor de Amortiguamiento

Los tres algoritmos de entrenamiento tienen un parámetro que puede variar durante las épocas:

SGD: alfa

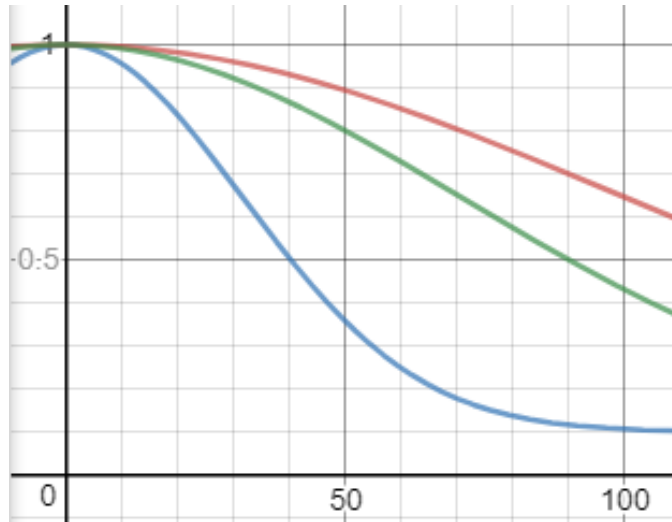
DE: h

PSO: c3

En la ecuación, ese parámetro es “v” y “rv” sería el valor verdadero usado en el entreno, eje Y.

El eje X, es la variable “t” que refiere a la iteración actual, de T iteraciones máximas.

$$rv = (0.9.e^{\frac{-t^2}{2.f.T^2}} + 0.1).v$$



En la gráfica se tomó:

v = 1

T = 100 iteraciones (épocas)

f (factor amortiguamiento) =

100 % rojo

50 % verde

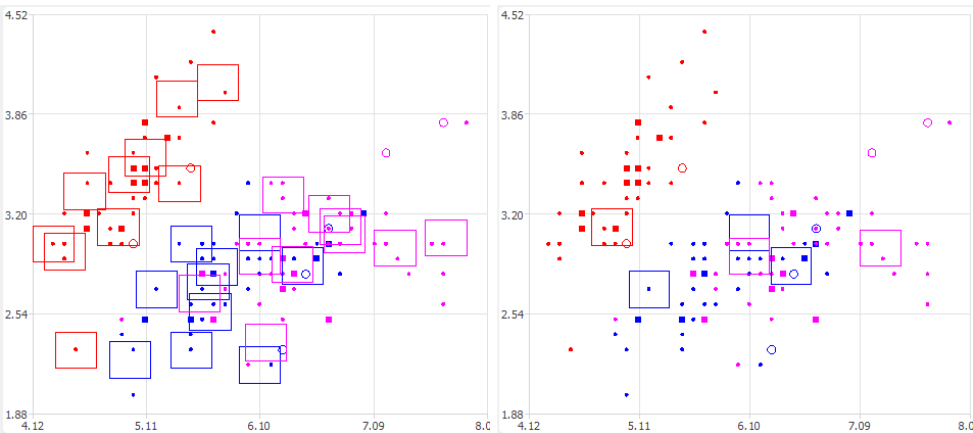
10 % azul

En el software un f = 0 % implica que no cambiará el parámetro: rv = v

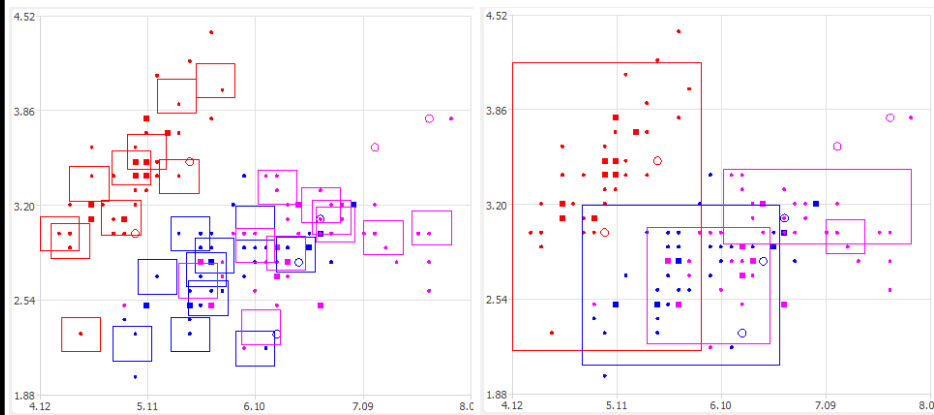
Algoritmos de Optimización

Eliminación de Dendritas: recorre todas las dendritas (hipercajas) y para cada una revisa el cambio en el error si está no estuviera; de cumplir con No cambiar el error más de cierta tolerancia, el cambio será permanente.

Este método se puede usar durante el entreno, en el software, selecciona de a una dendrita al azar.



Unión de Dendritas: recorre todas las dendritas (hipercajas) y para cada una revisa si se puede unir con otras de su misma clase, dando una hipercaja mayor; de cumplir con No cambiar el error más de cierta tolerancia, el cambio será permanente.



Hay matriz para:
Entreno
Validación
Testeo
y sus
combinaciones

Matrices de Confusión

Diagonal son los
verdaderos (V) de
la matriz, el resto
son falsos (F).

Predicciones hechas
por la red neuronal.

Porcentaje de los datos
de dicha clase que
fueron correctamente
clasificados.

Datos reales
aportados por el
set de patrones.

Media armónica
entre la Exactitud y
la Sensibilidad,
relaciona ambas
mediciones.

Probabilidad de que
la predicción
entregada por el
detector pertenezca
realmente a dicha
clase.

Porcentaje de los datos No pertenecientes
a dicha clase, que fueron correctamente
clasificados como No pertenecientes.

Porcentaje de datos
correctamente
clasificados.

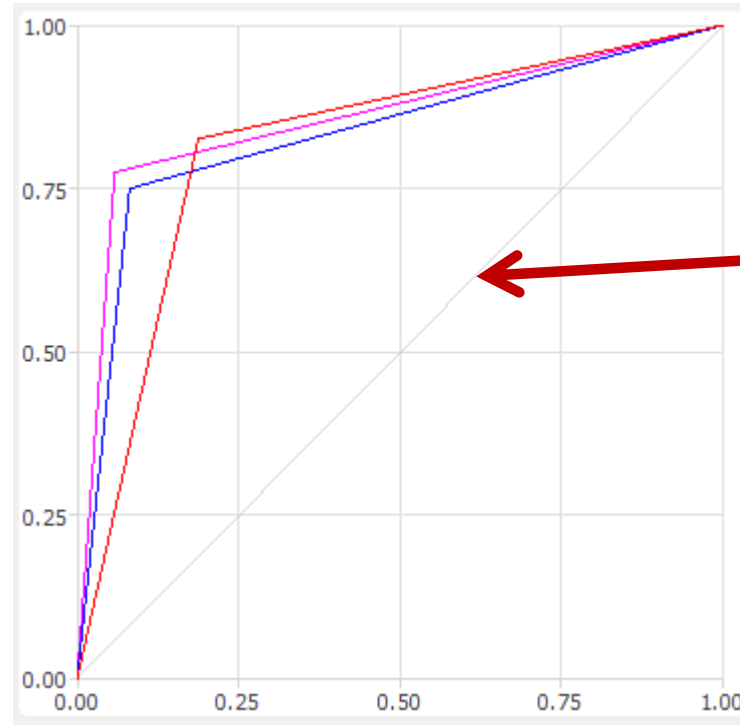
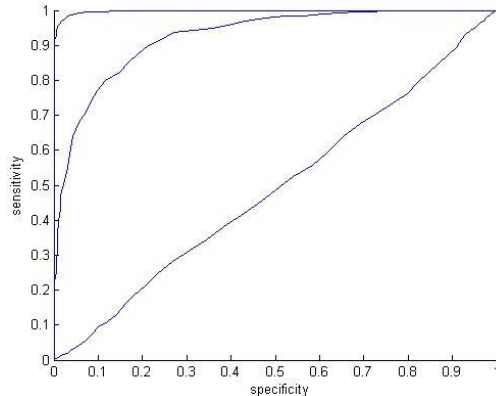
Grado de acuerdo entre
dos mediciones (reales vs
predichos) tomando en
consideración al azar.

	1	2	3	4	5	6	7
1	PREDICHOS →	P0	P1	P2	Total	Sensibili.	Valor-F
2	R0: ZancudoAgrio	39	5	2	46	84%	75%
3	R1: Azuladito	7	32	1	40	80%	78%
4	R2: Purpuroso	11	5	24	40	60%	71%
5	Total	57	42	27	126		
6	Exactitud	68%	76%	88%		Precisión	75%
7	Especificidad	77%	88%	96%		Kappa	0.628

ROC

El eje X es (1 - especificidad) y el eje Y es la sensibilidad, por tanto, esta gráfica relaciona las dos magnitudes; su punto óptimo es 0,1 (esquina superior izquierda).

La curva de región de convergencia debería lucir como la imagen de abajo.



En el software solo contamos con un punto por clase, así que eso es lo que puede verse en la imagen para 3 clases.

La línea gris diagonal significa un sistema aleatorio (50 % probabilidad), valores inferiores son clasificadores pésimos, pero reversibles si se invierten sus salidas (simetría en torno a la línea).

Gracias por usar SoftwareDMNN



<http://ojorcio.000webhostapp.com/>

ojorcio@gmail.com

https://www.dropbox.com/sh/plhbo1ornjah8j/b/AAAOdaSe5JArLE1XRo--Eh_7a?dl=0

