Resolución TP2

Clasificador basado en Ley de Aprendizaje Supervisado

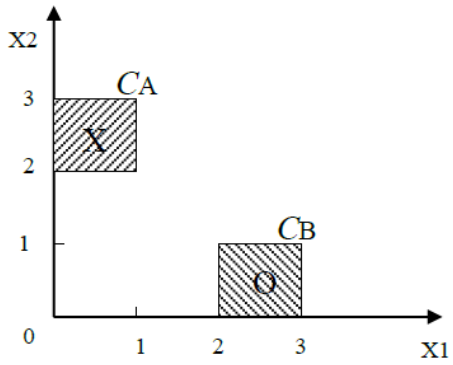
Alumno: Salomón Nicolás

Curso: Inteligencia Artificial aplicada a la Identificación y Control

Profesor: Dr. Ing. H. Daniel Patiño

Año: 2022

1. **Desarrollar un perceptron que realice la tarea de Clasificación capaz de clasificar los Clusters A y B. Las Clases se muestran en la siguiente figura:**



a) Realizar un diseño analítico para encontrar las sinapsis del perceptrón y graficar el “hiperplano” de separación.

b) Generar el código para el diseño automático del clasificador basado en un aprendizaje supervisado. Emplear cualquiera de las leyes de aprendizaje estudiadas: gradiente descendente, Levenberg-Marquardt, o Mínimos Cuadrados Recursivo.

c) Mostrar en gráficas la evolución de índices que monitorean el aprendizaje: e(k) y los parámetros Wi(k).

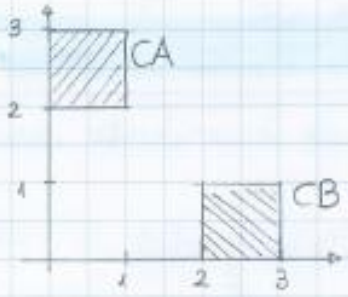
El código de desarrollo del presente trabajo se encuentra adjunto al mismo (TP2 - Perceptrón.ipynb)

La resolución analítica se desarrolló de la siguiente forma:

IA Aplicada a la Identificación y Control

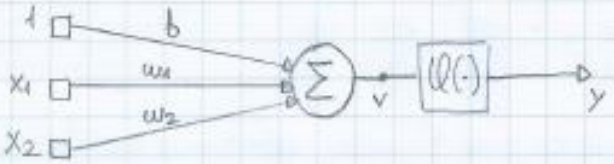
TP N°2 - Diseño de un clasificador

Desarrollar un perceptrón que realice la tarea de clasificación, capaz de clasificar los Clusters A y B



a)- Realizar un diseño analítico para encontrar la sinápsis del perceptrón y graficar el hiperplano de separación

El perceptrón a desarrollar tendrá la siguiente estructura:



$$v = w_1 x_1 + w_2 x_2 + b$$

$$y = \varphi(v)$$

Existen muchas funciones de activación posibles, pero para un desarrollo analítico consideraremos la función escalón:

$$\varphi(v) = \begin{cases} 1 & \text{si } v > 0 \\ 0 & \text{si } v \leq 0 \end{cases}$$

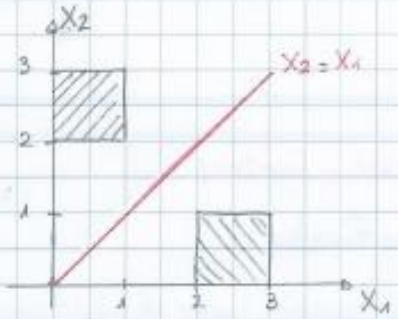
En base a esto:

$$y(w, b, x) = w_1 . x_1 + w_2 x_2 + b$$

Para despejar x_2 en función del resto de variables y poder graficar el hiperplano de separación, tomaremos $y = 0$

$$0 = w_1 . x_1 + w_2 . x_2 + b \Rightarrow x_2 = \underbrace{-\frac{w_1}{w_2}}_{\text{pendiente}} . x_2 - \underbrace{\frac{b}{w_2}}_{\text{ordenada al origen}}$$

Tomando el plano $x_1 = x_2 \Rightarrow -\frac{w_1}{w_2} = 1$ y $-\frac{b}{w_2} = 0 \Rightarrow$ $b = 0$, $w_1 = 1$, $w_2 = -1$



Cabe destacar que no es el único hiperplano que se podría trazar para separar los 2 clusters.

b) La programación para el diseño automático puede verse en el archivo adjunto, desarrollado en lenguaje Python y Jupyter Notebook.

En dicho diseño se empleo el gradiente descendente, considerando el error cuadrático instantaneo.

$$e(k) = d(k) - y(k) \quad ; \quad C(k) = \frac{1}{2} \cdot e(k)$$

donde $d(k)$ es el valor deseado, $y(k)$ es la salida de la red neuronal y $C(k)$ el funcional de costo

Para aplicar el gradiente descendente necesito obtener $\frac{\partial C}{\partial w}$, recordando que

$$y(w, b, x) = \varphi(v) = \varphi(w_1 x_1 + w_2 x_2 + b) \quad , \quad \frac{\partial C}{\partial w} = e(k) \cdot \frac{\partial e}{\partial w}$$

$$\frac{\partial e}{\partial w} = -\frac{\partial y}{\partial w} = -\frac{\partial y}{\partial \varphi} \cdot \frac{\partial \varphi}{\partial v} \cdot \frac{\partial v}{\partial w} = -\frac{\partial \varphi}{\partial v} \cdot \frac{\partial v}{\partial w}$$

Empleando como función de activación una función continuamente diferenciable como una sigmoide:

$$\varphi(v) = \frac{1}{1 + e^{-a \cdot v}}$$

$$\frac{\partial e}{\partial w_1} = a \cdot \varphi(v)(1 - \varphi(v)) \cdot x_1 \qquad \frac{\partial e}{\partial b} = -a\, \varphi(v)(1 - \varphi(v))$$

$$\frac{\partial e}{\partial w_2} = a\, \varphi(v)(1 - \varphi(v)) \cdot x_2$$

Por lo tanto, la actualización de pesos será la siguiente:

$$w_1(k+1) = w_1(k) - \mu \frac{\partial C}{\partial w_1} = w_1(k) - \mu \frac{\partial e}{\partial w_1} \cdot e(k)$$

$$w_2(k+1) = w_2(k) - \mu \frac{\partial C}{w_2} = w_2(k) - \mu \frac{\partial e}{\partial w_2} \cdot e(k)$$

$$b(k+1) = b(k) - \mu \frac{\partial C}{\partial b} = b(k) - \mu \frac{\partial e}{\partial b} \cdot e(k)$$

Donde μ es el coeficiente de aprendizaje, relacionado con la velocidad de aprendizaje durante el entrenamiento

Por otro lado, se desarrolló un perceptrón como se puede apreciar en el código adjunto a este informe. El mismo se entrenó empleando el algoritmo de Back-propagation en conjunto con el Gradiente Descendente, durante 10000 iteraciones, con un tiempo total de 0.41 [s].

El resultado del entrenamiento puede apreciarse en la Figura 1.

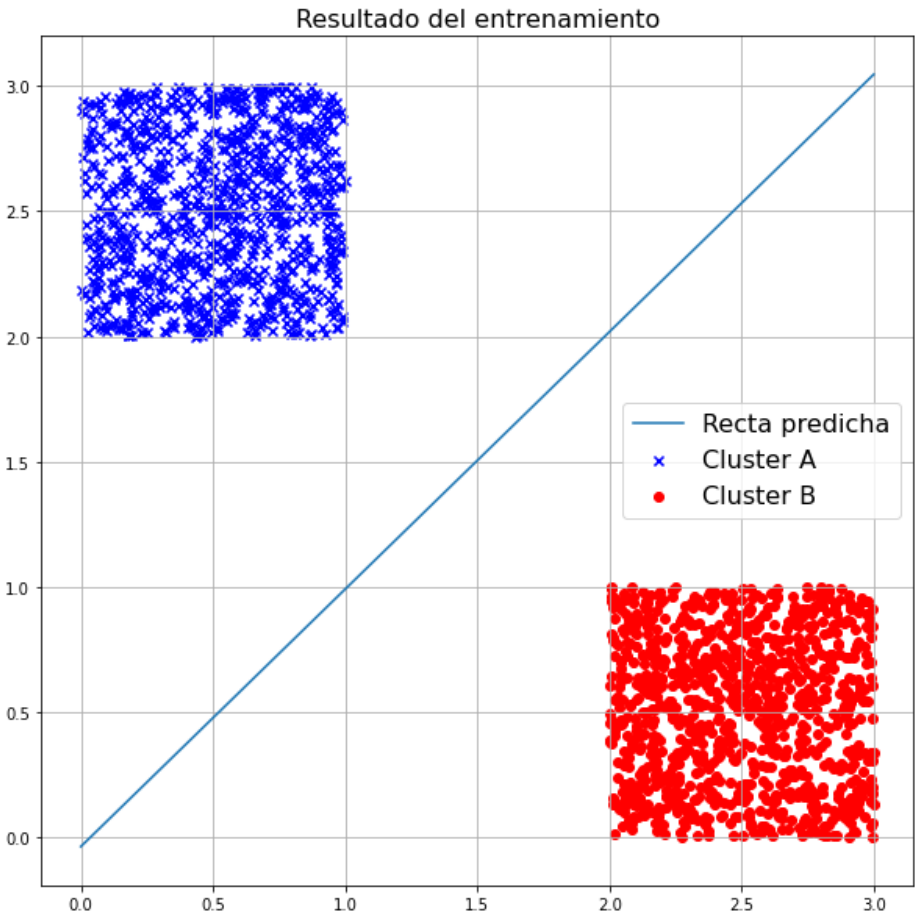


Figura 1. Resultado del entrenamiento.

Se puede observar como la clasificación es perfecta al observar la línea que divide ambos clusters.

Por otro lado, también se observan en las Figuras 2 y 3 las variaciones de los pesos y del error respectivamente.

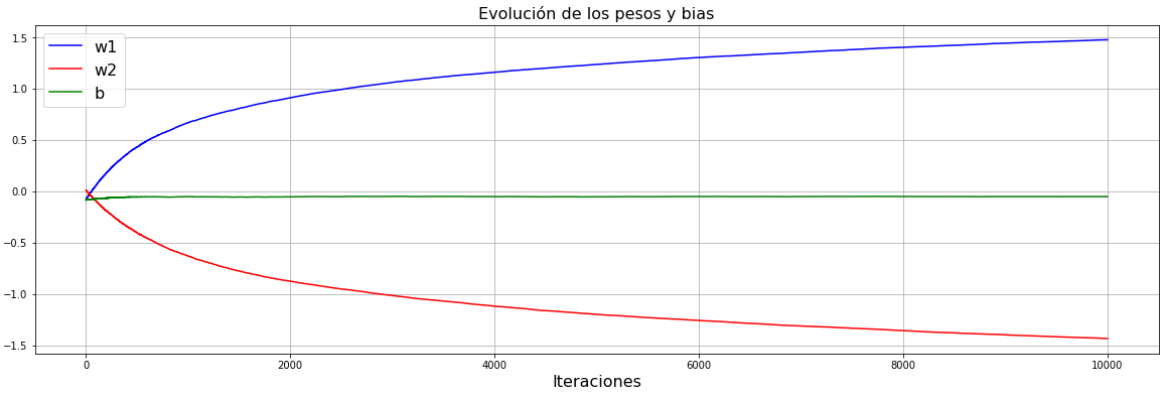


Figura 2. Variación de los pesos y bias.

Se observa como los pesos y bias tienden a su valor final sin ningún sobresalto y de manera progresiva, para tender a estabilizarse hacia el final, indicando un correcto proceso de aprendizaje, sin sobreajuste. Esto es confirmado al observar la Figura 3 con la variación del error.

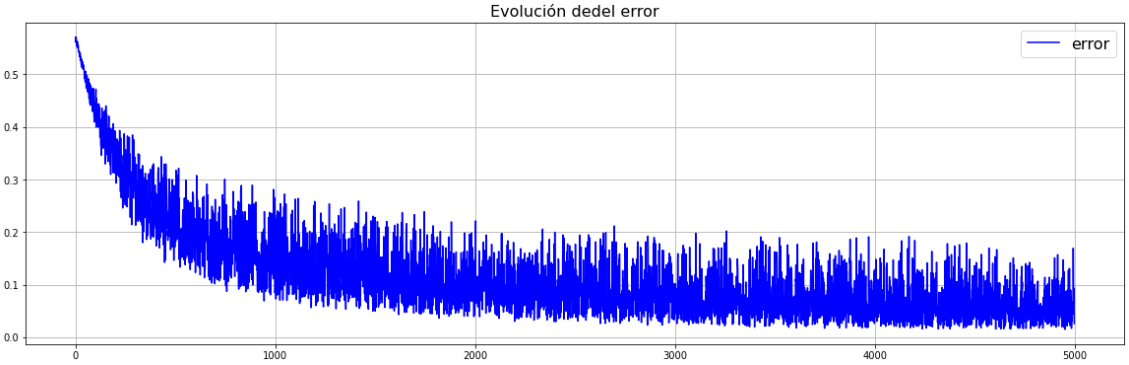


Figura 3. Variación del error.

Se observa un error continuamente descendiente que comienza a estabilizarse hacia el final del entrenamiento.

Por último, para comprender de mejor manera el proceso de aprendizaje, se muestran los resultados intermedios obtenidos durante el proceso de aprendizaje.

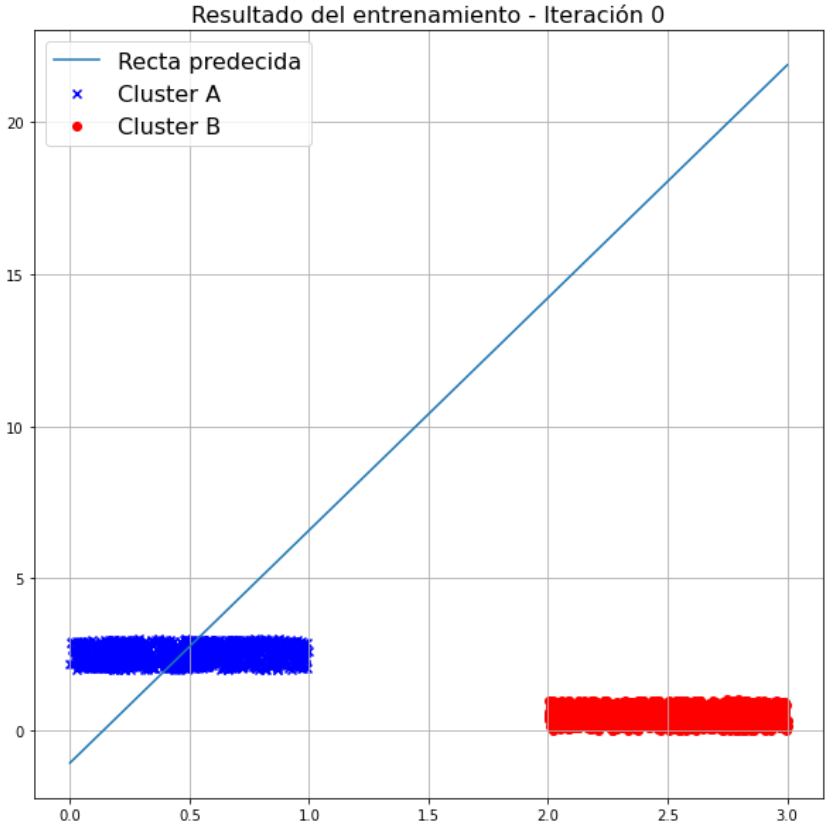


Figura 4. Resultado de la primera iteración.

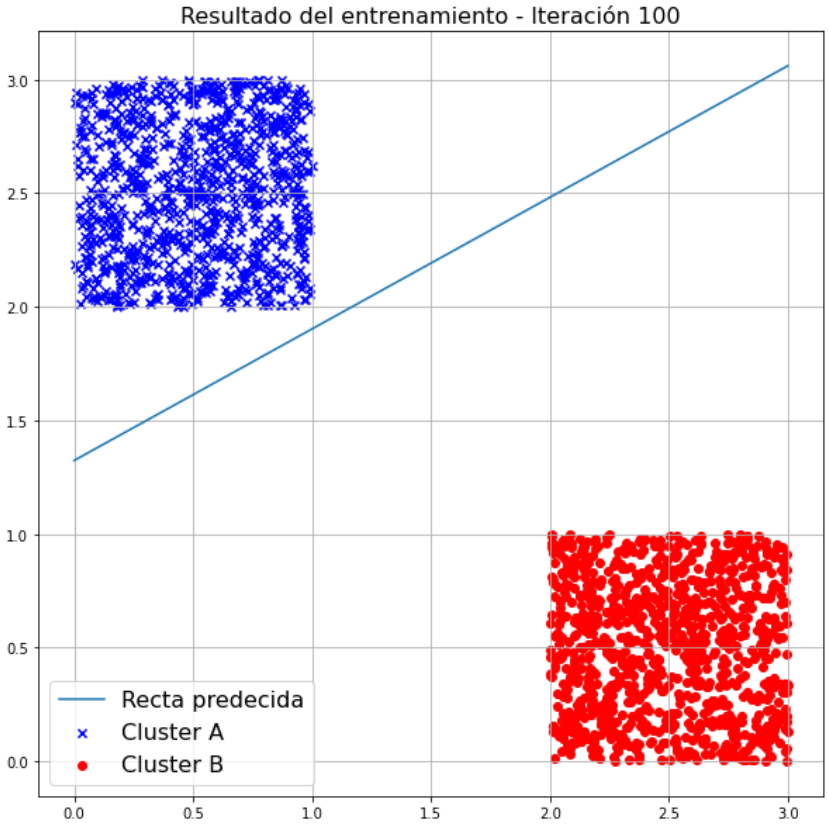


Figura 5. Resultado de la iteración Nº 100.

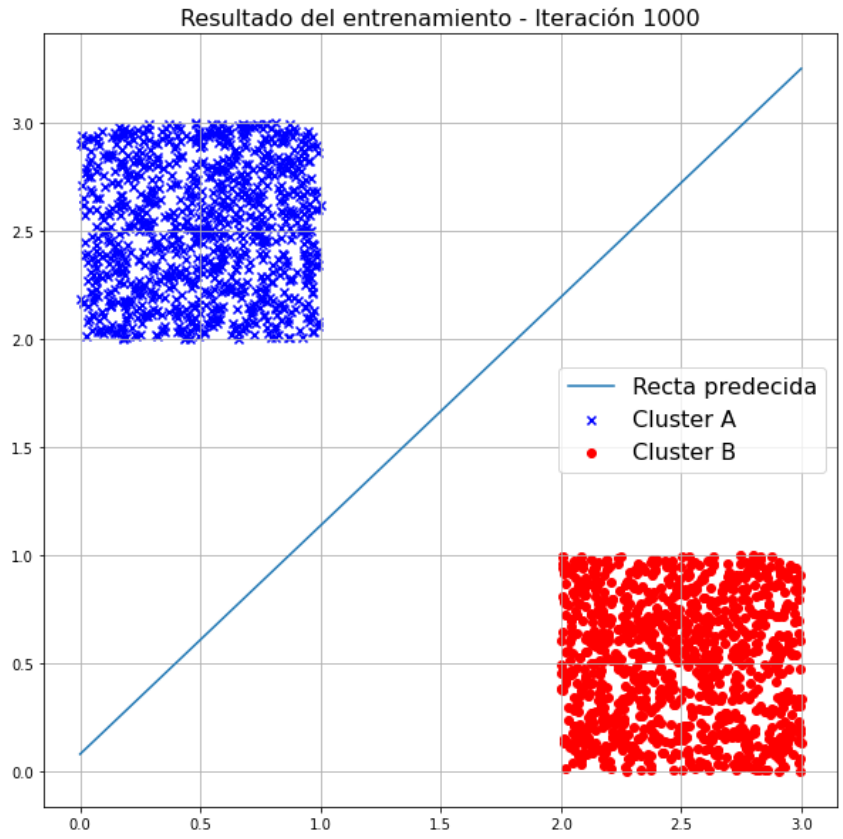


Figura 6. Resultado de la iteración Nº 1000.

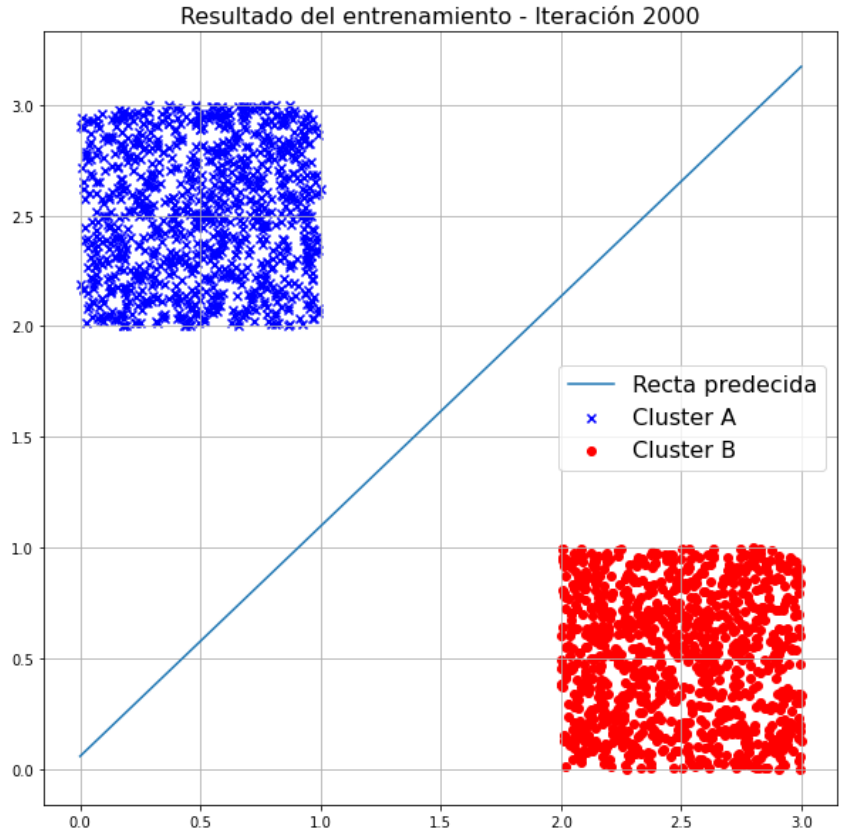


Figura 7. Resultado de la iteración Nº 2000.

Se observa como la curva que delimita las zonas de decisión de la red neuronal va ajustándose a medida que avanzan las iteraciones hacia su valor óptimo, aquel que dista lo mismo hacia los centros de cada cluster. Esto nos muestra un correcto proceso de aprendizaje.

Por último, se generó un set de pruebas aleatorios para que el algoritmo los clasifique, arrojando los resultados de la Figura 8.

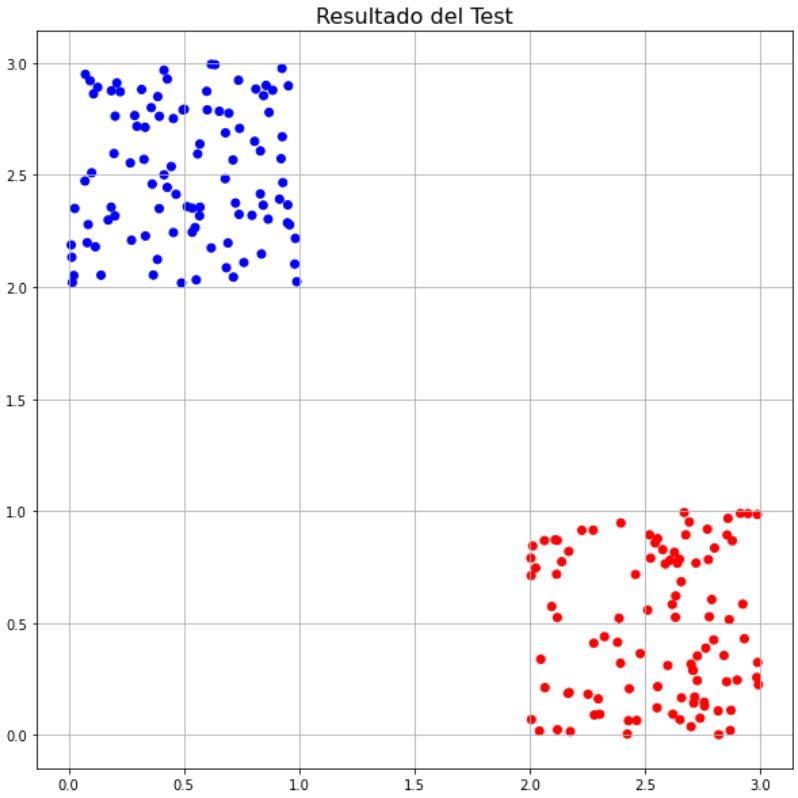


Figura 8. Resultado sobre un dataset de prueba.

Se observa una clasificación correcta para todos los puntos del set de pruebas aleatorio generado, lo que indica un correcto funcionamiento de la red.