UNIVERSIDAD DE GUANAJUATO

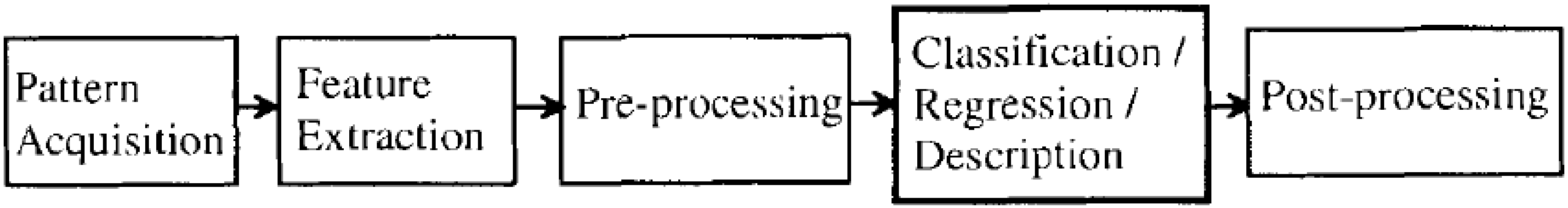
Inteligencia Artificial

Agosto-Diciembre 2021
Lunes y Miércoles
10:30 – 12:00

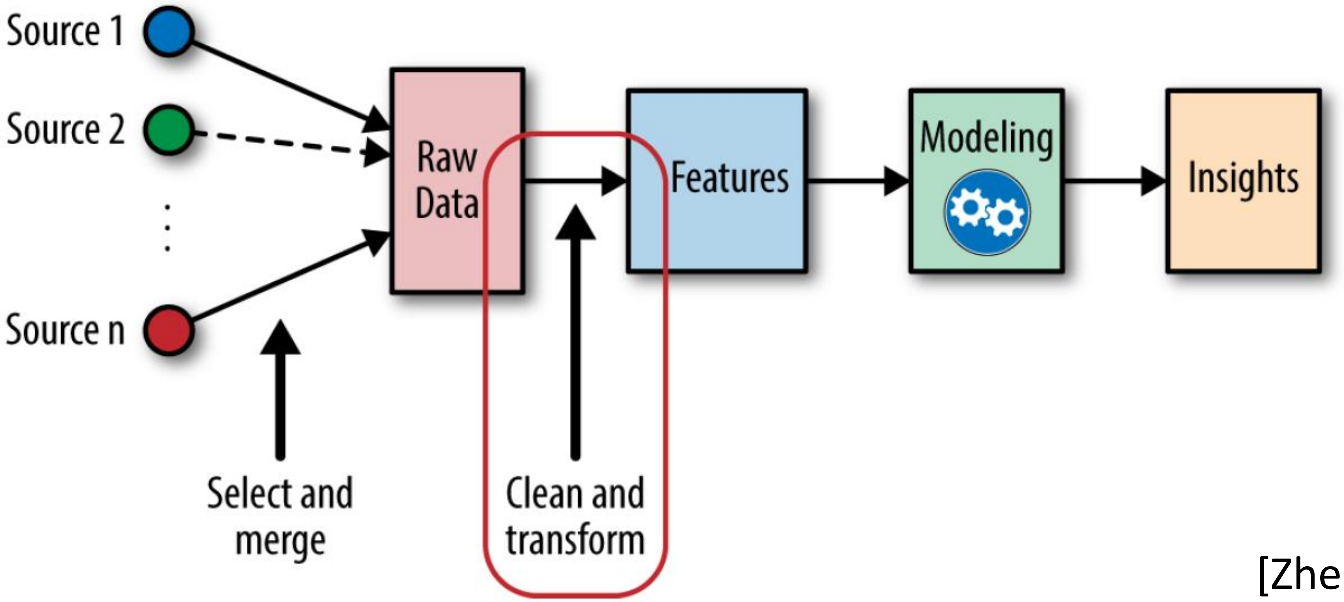
Contenido Unidad II

- ¿Que es el Reconocimiento de Patrones (RP)?
- Taxonomía de Modelos
- Tareas Supervisadas
- Tareas No Supervisadas

- “Disciplina científica que trata con métodos para la descripción y clasificación de objetos” [Marques de Sá, 2001]
- ¿Aprendizaje Máquina? “Ajusta modelos matemáticos a datos para derivar conocimiento o hacer predicciones” [Zheng,2018]. “Es usar las características adecuadas para construir los modelos correctos con el fin de lograr las tareas correctas” [Flatch,2012]
- … ¿Aprendizaje profundo? Revisar

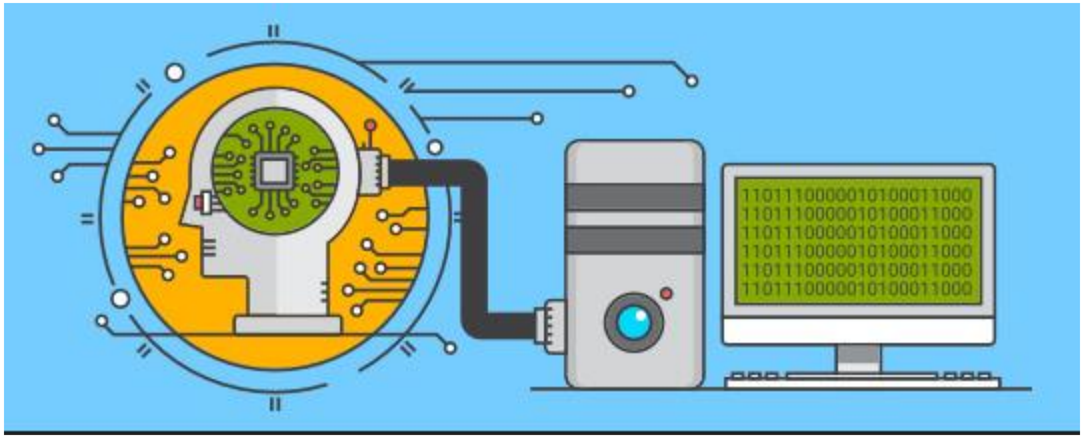


[Marques de Sá,2001]

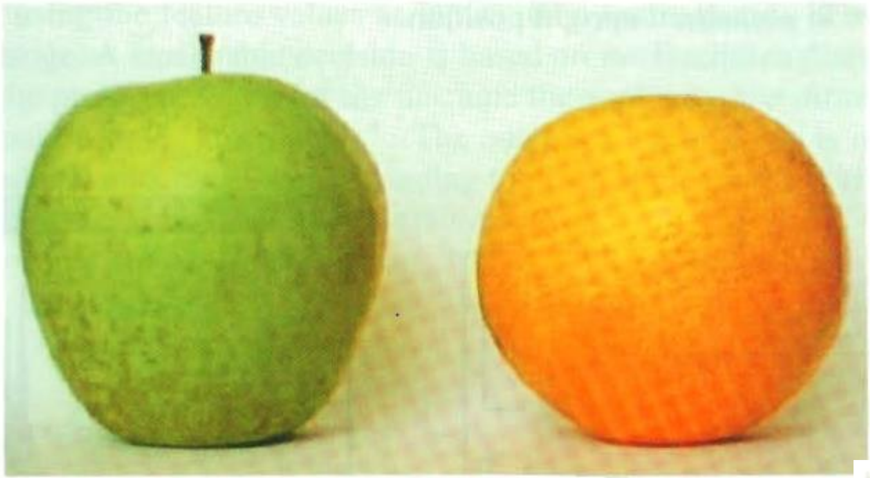


[Zheng,2018]

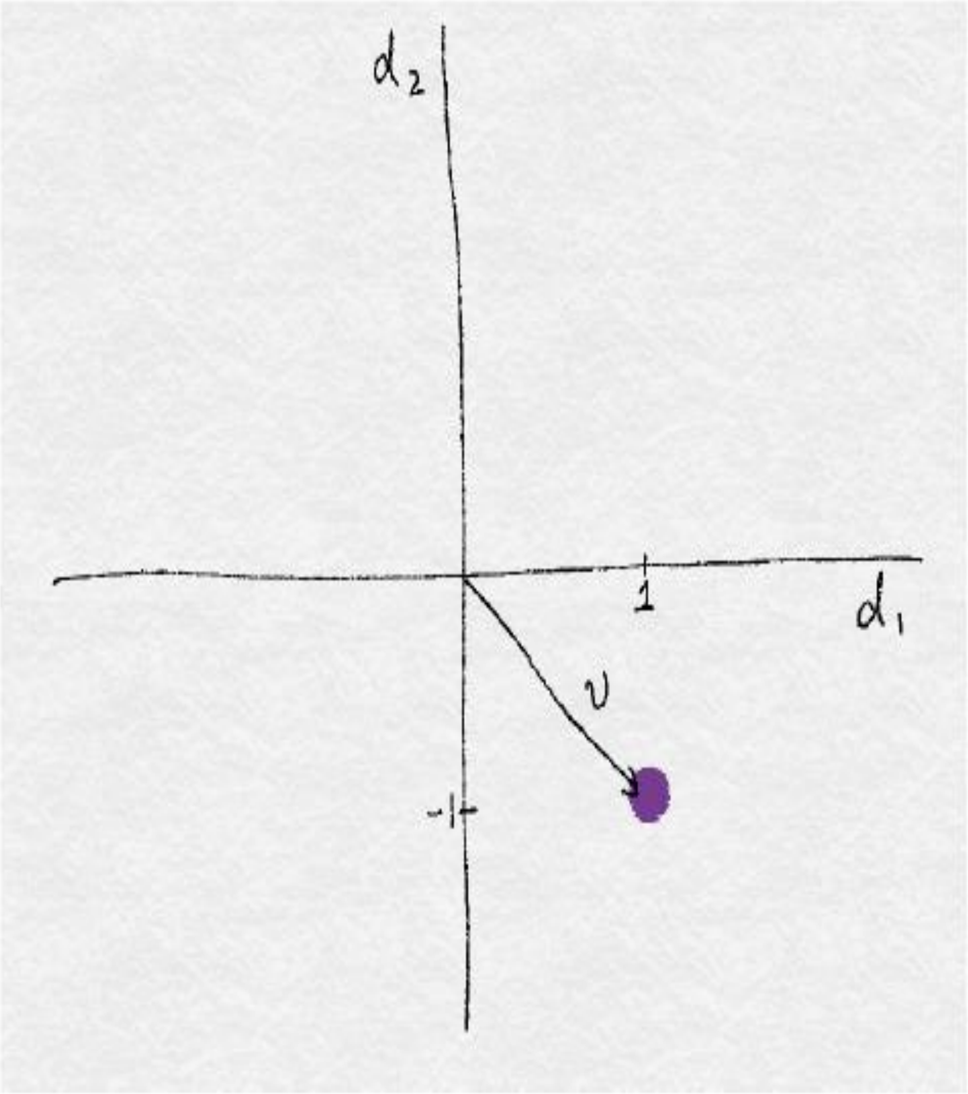
- Tareas Supervisadas
  - Clasificación
  - Regresión
- Tareas No supervisadas
  - Agrupamiento



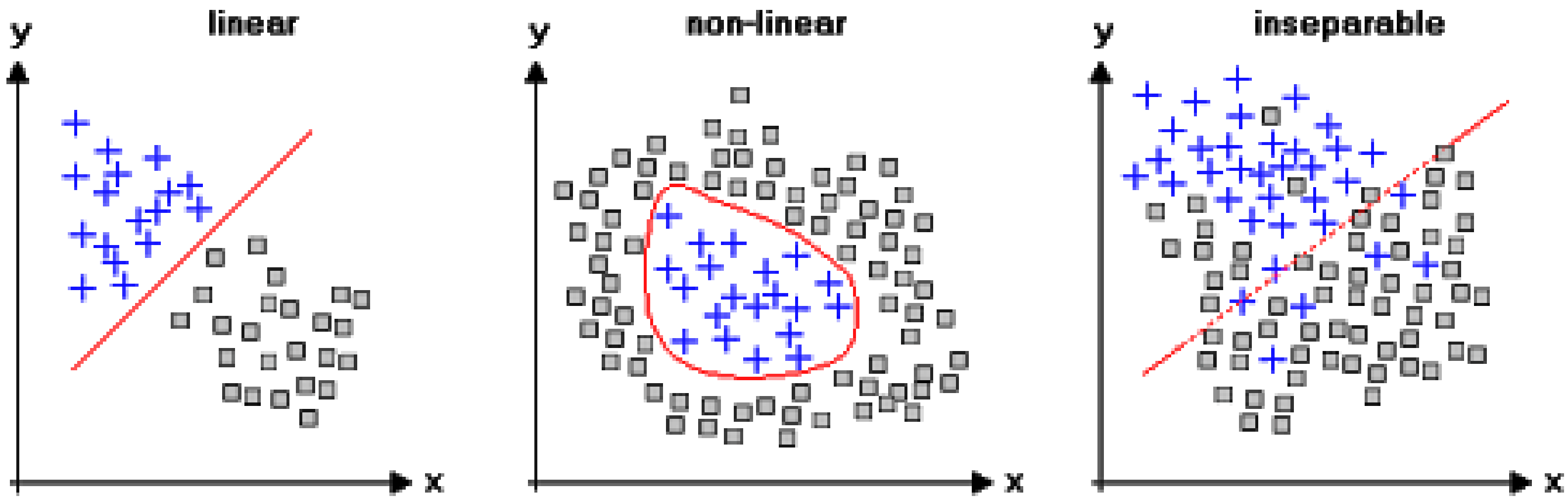
- Cualitativas
- Cuantitativas



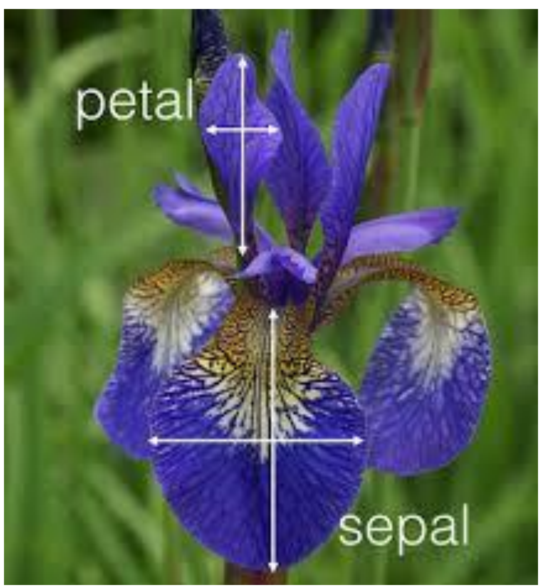
¿Vector de Características y Espació de Características?



Posibles Espacios de Características



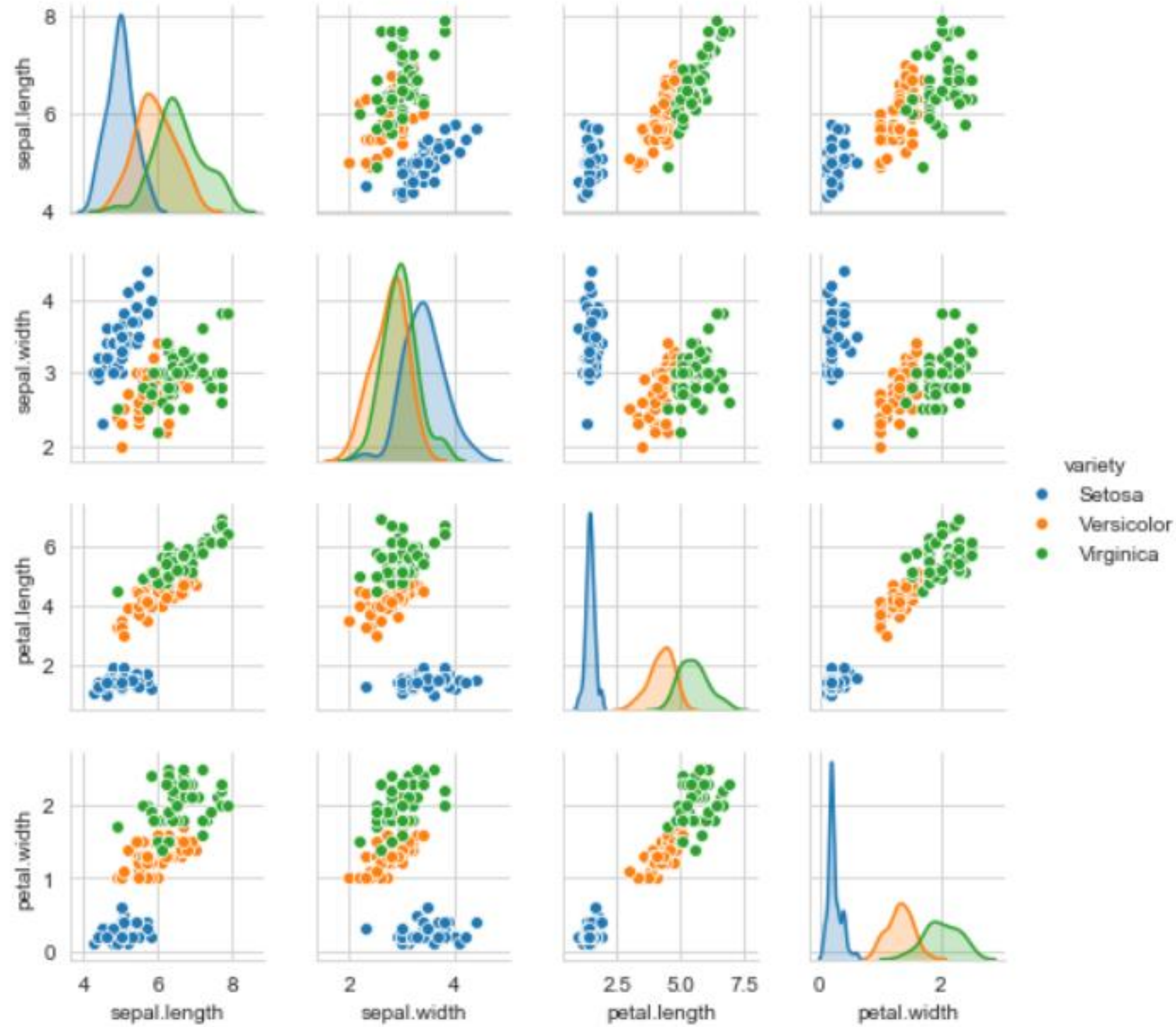
Ejemplo Clasificación: Iris Data Set
[https://archive.ics.uci.edu/ml/datasets/iris]



Vector de Características

$$x = [sépalo_{largo}, sépalo_{ancho}, pétalo_{largo}, pétalo_{ancho}]$$

Pairs Plot del Iris Data Set



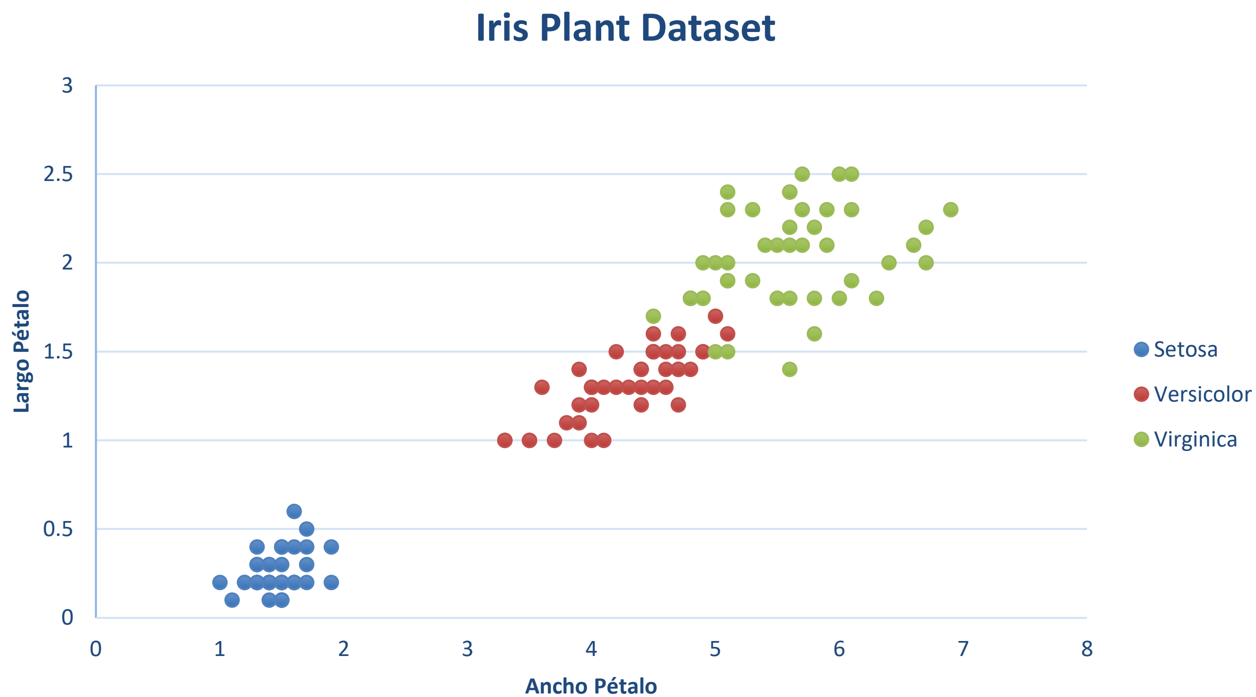
- Taxonomía de Clasificadores según [Knox,2018]:
  – Métodos de Prototipo
  – Métodos de Probabilidad
  – Regresión Logística
  – Redes Neuronales
  – Árboles de Clasificación
  – Máquinas de Vector Soporte

- Taxonomía de Clasificadores según [Friedman,199]:
  - Basada en Distancia
  - Enfoque Estadístico
  - Redes Neuronales

Métodos de Prototipos

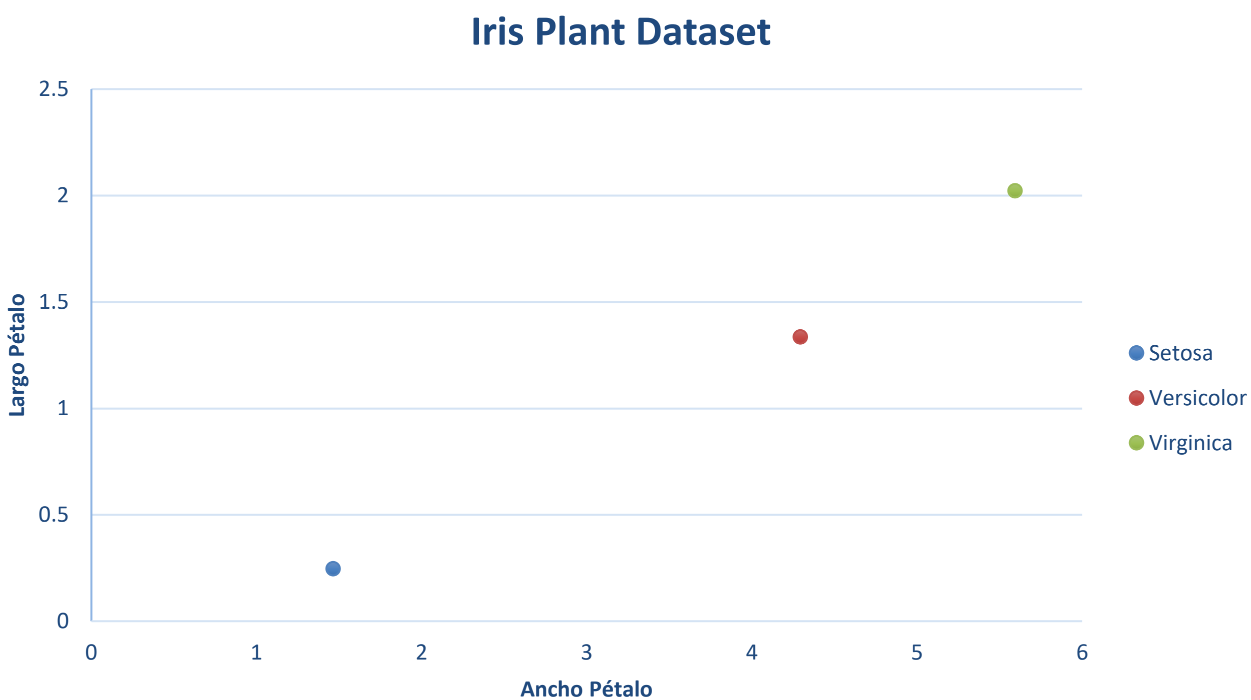
- Clasificador de Mínima Distancia
- Clasificador del K Vecino Más Cercano

- Para un problema de m clases con patrones d-dimensionales y un conjunto de patrones de entrenamiento $\{(x_i, y_i)\}$ de N patrones $(i = \{1, \ldots N\})$.



UNIVERSIDAD DE GUANAJUATO

- Se obtienen un vector prototipo por cada clase (ej. Vector promedio o centro de masa de clase): $x_j'; j = \{1, \ldots, m\}$



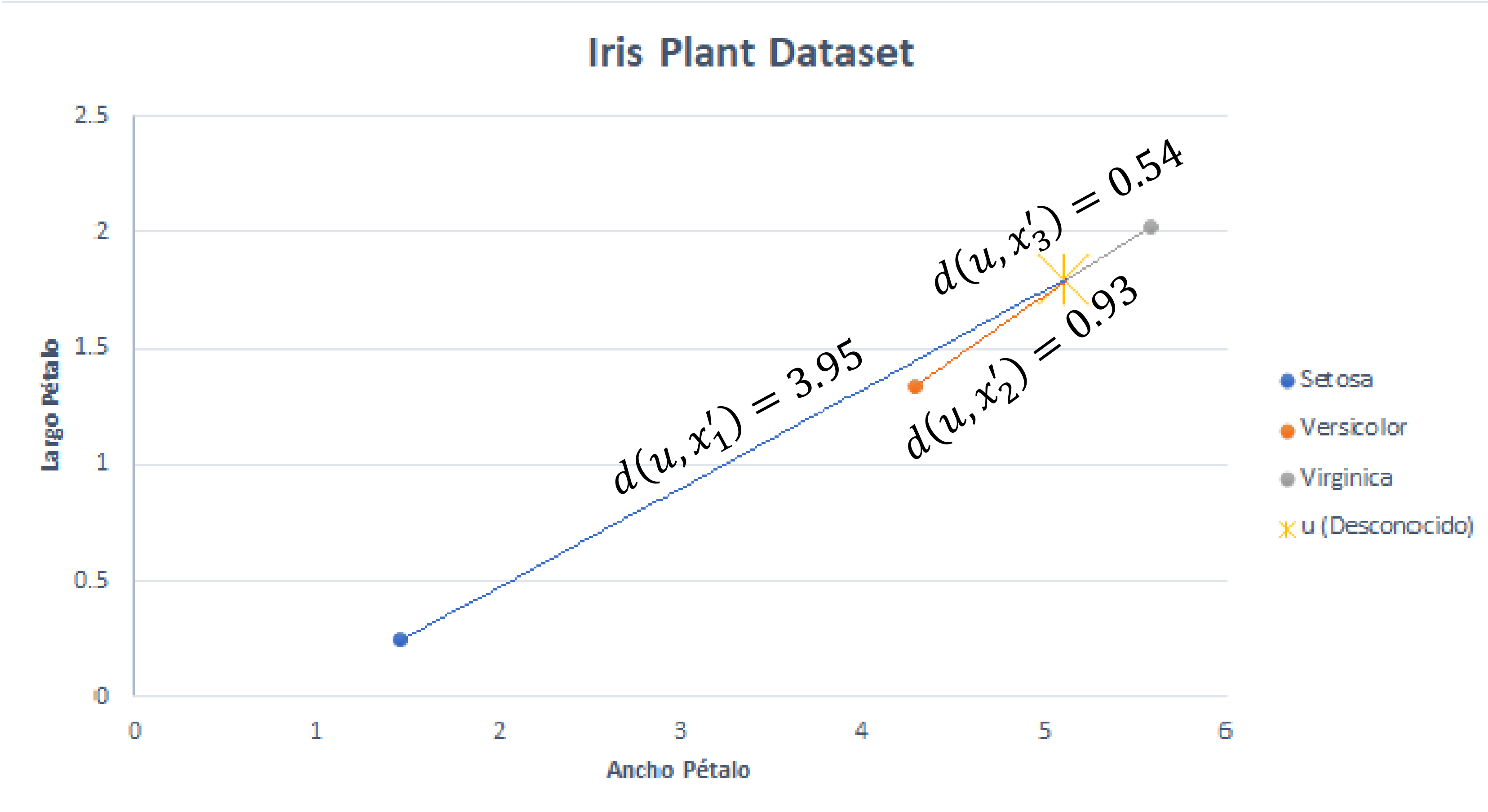
- Para clasificar un patrón u, se calcula la distancia de u con respecto de cada vector prototipo x_j' y se obtiene j_c que indica el índice del vector prototipo con el que u tuvo la menor distancia (está más cercano).
- Distancia Euclídea

$$d(u,v) = \sqrt{\sum_{i=1}^{d}(u_i - v_i)^2}\ ; i = \{1, \ldots, d\}$$

- Asignación de clase

$$j_c = \min_{1 \le c \le m}\{d(u, x_c')\}$$

Clasificador de Mínima Distancia: Paso 3



- u es clasificado en la clase 3, que corresponde a virginica; lo cual es correcto, dado que esta es su etiqueta en el registro original (registro 150).

- Dados los siguientes vectores prototipo:
  - $x_1' = (1.46, 0.25)$ – Setosa
  - $x_2' = (4.29, 1.34)$ – Versicolor
  - $x_3' = (5.59, 2.02)$ – Virginica
- Clasificar los siguientes patrones
  - $u_1 = (1.4, 0.4)$
  - $u_2 = (1.4, 0.2)$
  - $u_3 = (4.2, 1.2)$
  - $u_4 = (5.1, 1.6)$
  - $u_5 = (5.1, 1.9)$
  - $u_6 = (5.2, 2.3)$

- Dada la siguiente información: u_1 y u_2 son setosa, mientras que u_3 y u_4 son versicolor y por último u_5 y u_6 son virginica; realice la siguiente tabla de conteo (**Matriz de confusión**). Cada renglón indica la clase verdadera y la columna la clase donde fue clasificado

$M =$

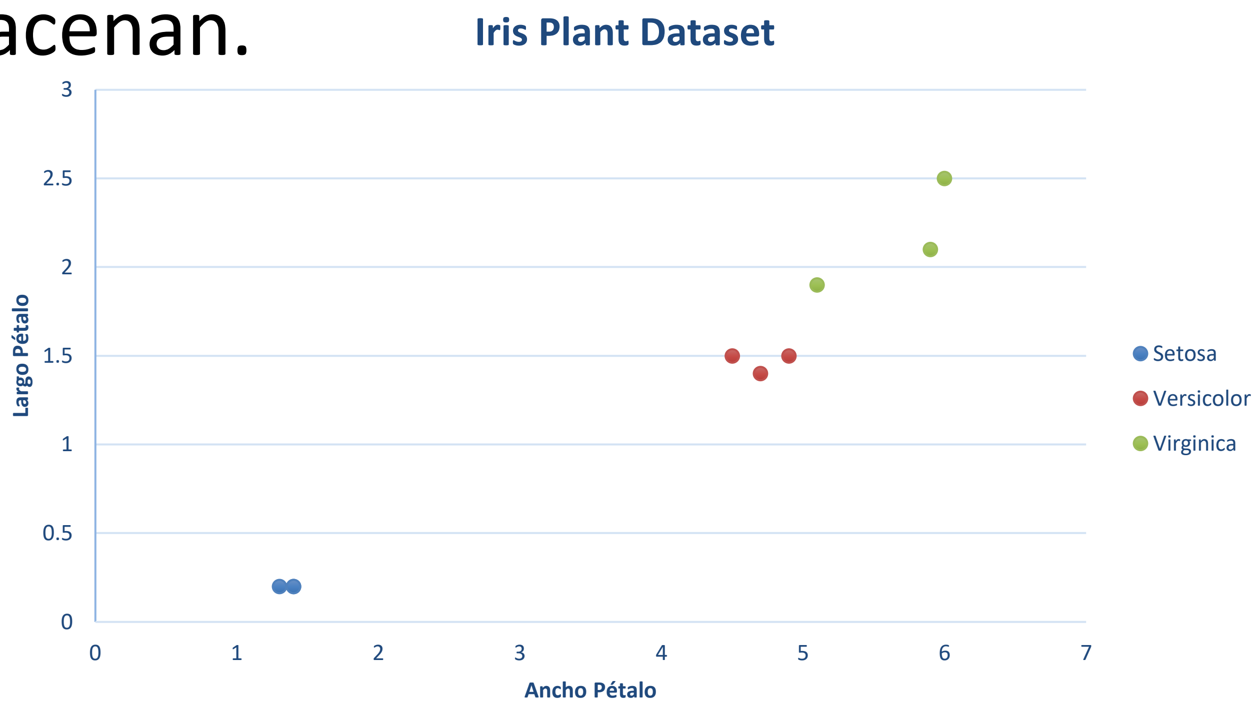
	Setosa	Versicolor	Virginica
Setosa			
Versicolor			
Virginica			

- Calcule la exactitud (Accuracy) del clasificador, mediante la matriz de confusión. Excluyendo los encabezados y columna de clase, la exactitud se calcula como la suma de los elementos en la diagonal principal dividida
- entre la cantidad de patrones de entrenamiento.

$$M = \begin{bmatrix} m_{11} & \cdots & m_{1m} \\ \vdots & \ddots & \vdots \\ m_{m1} & \cdots & m_{mm} \end{bmatrix}$$

$$Acc = \frac{\sum_{i=1}^{m} m_{ii}}{N}$$

- Para un problema de m clases con patrones d-dimensionales, se proporcionan: k (entero positivo, de preferencia impar) y un conjunto de patrones de entrenamiento $\{(x_i, y_i)\}$ de N patrones ($i = \{1, \ldots N\}$); los cuales se almacenan.

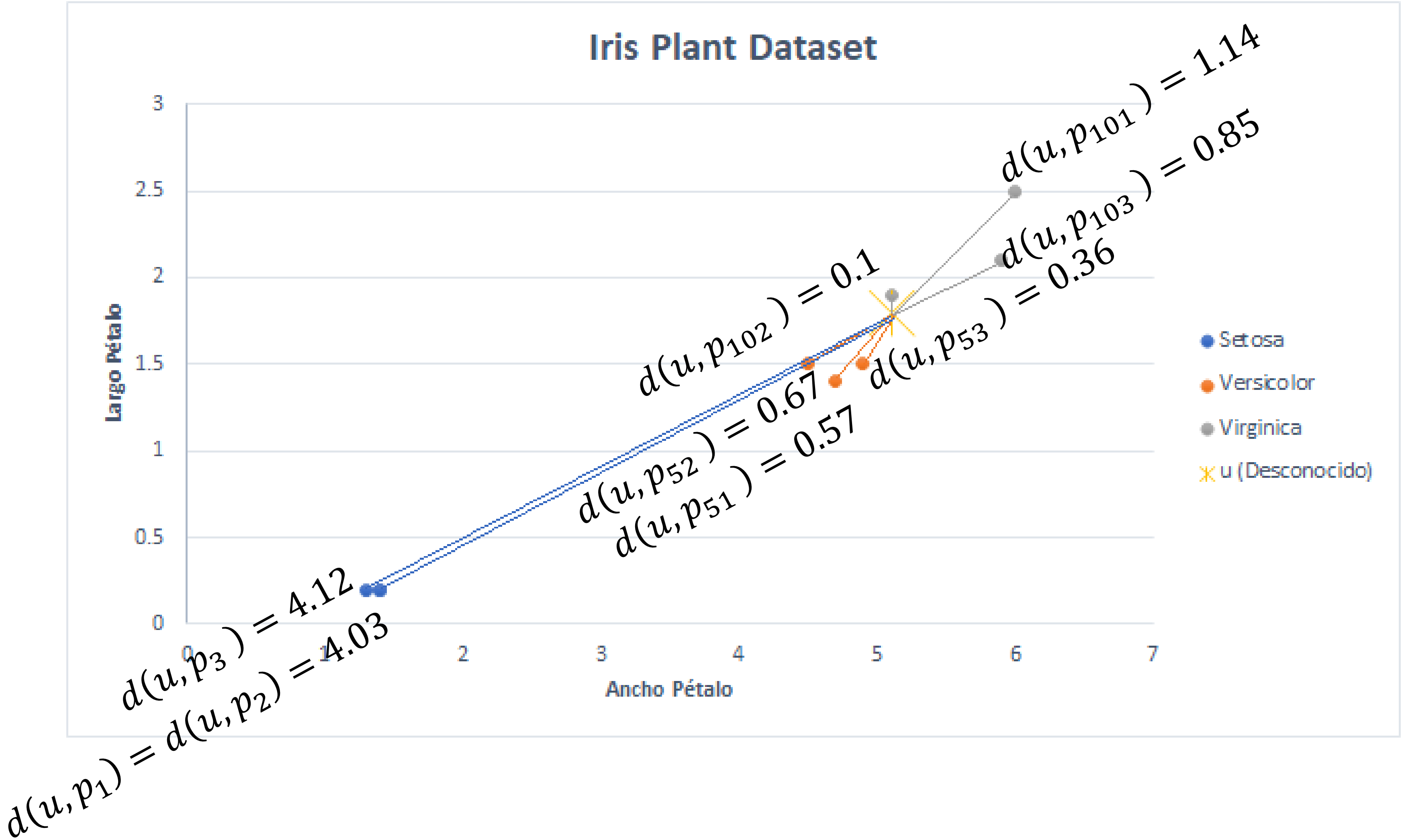


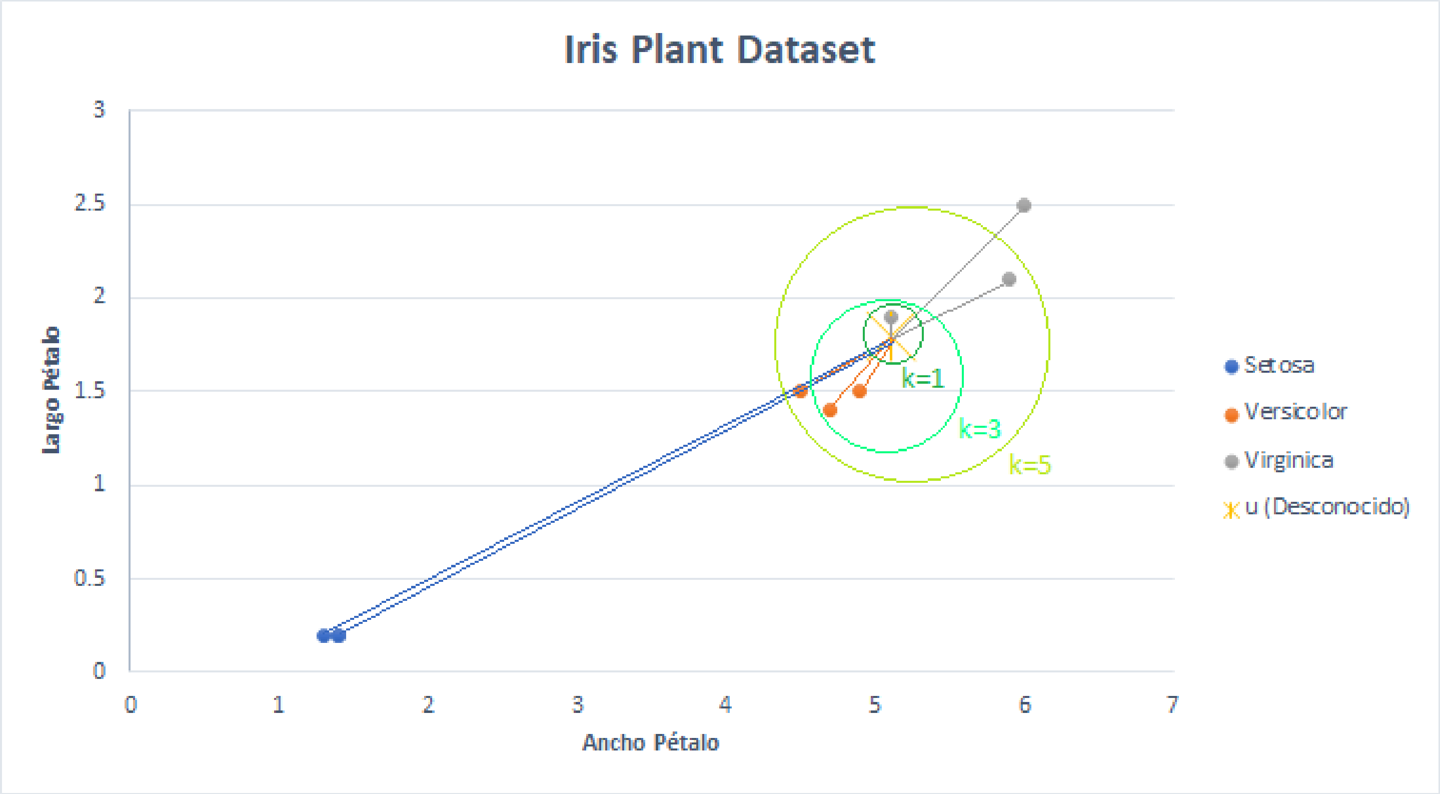
- Para clasificar un patrón u, se calcula y almacenan las distancias de u con respecto de cada vector en el conjunto de entrenamiento $\{(x_i, y_i)\}$; podemos generar la lista $\mathrm{D} = [(d(u, x_i), y_i)]$.
- Distancia Euclídea

$$d(u, v) = \sqrt{\sum_{i=1}^{d} (u_i - v_i)^2}\ ; i = \{1, \ldots, d\}$$

- Las distancias en D se ordenan de menor a mayor y se conservan las k primeras; $\mathrm{D} = \left[\left(d\left(u, x_j\right), y_j\right)\right]; j = \{1, \ldots, k\}$.
- Asignación de clase: se realiza un conteo de ocurrencia de los valores en y_j (como un histograma) y se asigna y_0 con el valor de la etiqueta de clase con más ocurrencias.

Clasificador del K Vecino Más Cercano (KNN)





- Si $k = 1$, entonces u es clasificado como virginica
- Si $k = 3$, entonces u es clasificado como versicolor
- Si $k = 5$, entonces u es clasificado como versicolor

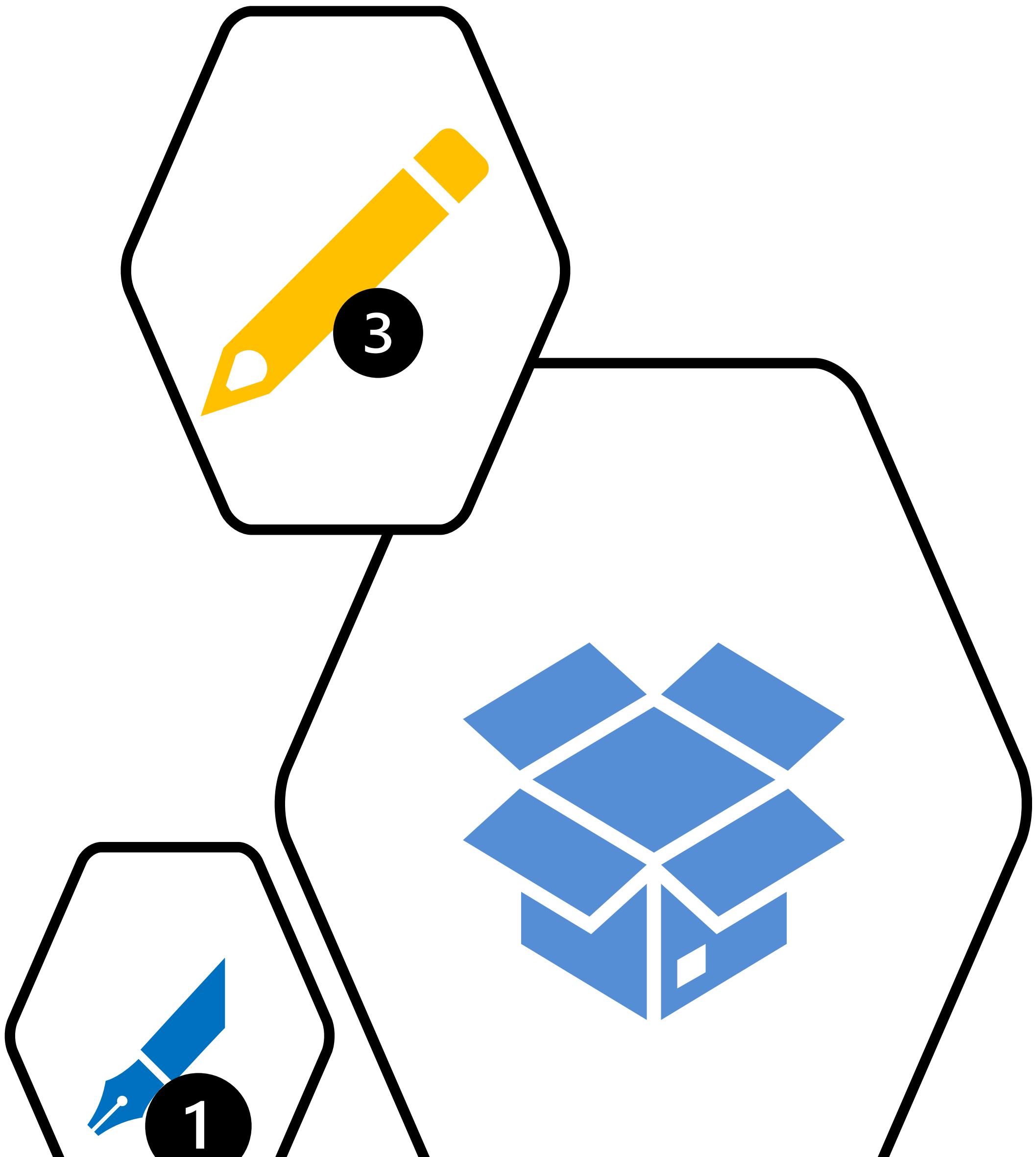
- Dados los siguientes vectores como conjunto de entrenamiento:
  - $x_1 = (1.4,0.2)$ – Setosa
  - $x_2 = (1.4,0.2)$ – Setosa
  - $x_3 = (1.3,0.2)$ – Setosa
  - $x_4 = (4.7,1.4)$ – Versicolor
  - $x_5 = (4.5,1.5)$ – Versicolor
  - $x_6 = (4.9,1.5)$ – Versicolor
  - $x_7 = (6.0,2.5)$ – Virginica
  - $x_8 = (5.1,1.9)$ – Virginica
  - $x_9 = (5.9,2.1)$ – Virginica
- Clasificar los siguientes patrones usando KNN con $k = 1$ y $k = 3$
  - $u_1 = (1.4,0.4)$
  - $u_2 = (5.1,1.6)$
  - $u_3 = (5.2,2.3)$
- Obtenga la exactitud de los ambos clasificadores.

Métodos de Probabilidad

- Clasificador de Bayes Ingenuo

¿Un clasificador basado en probabilidad?

- Experimento: Sin observar, extraer una herramienta de escritura de la caja y “adivinar” que tipo de herramienta es.
  - *Toda herramienta tiene la misma oportunidad de ser seleccionada*

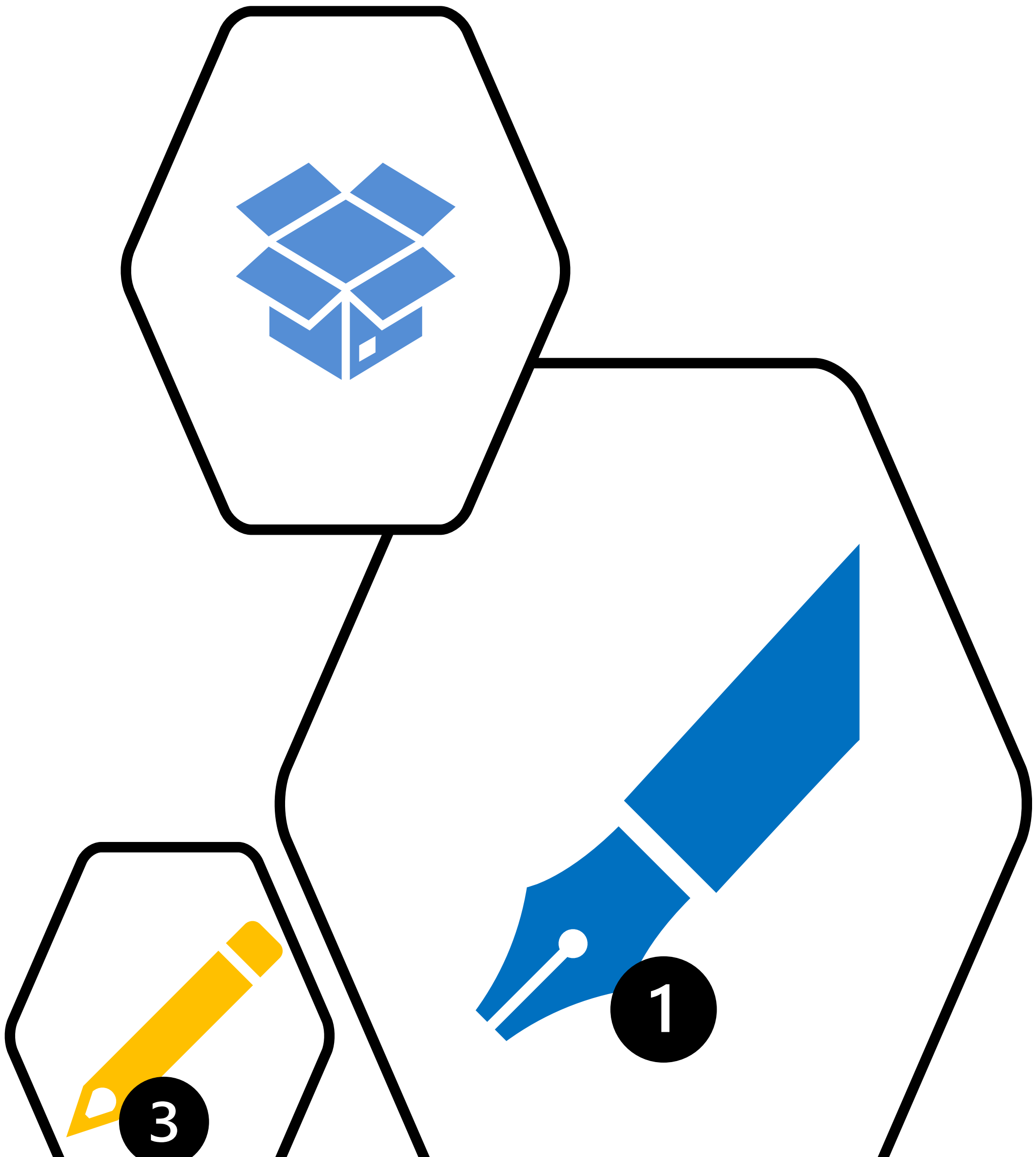


Clasificador Bayes Ingenuo: Preámbulo

¿Un clasificador basado en probabilidad?

- Sea Y una variable aleatoria para las herramientas
  - $P(Y = l) = {}^{3}/_{4}$
  - $P(Y = p) = {}^{1}/_{4}$
- ¿Podemos proponer un un clasificador en las probabilidades que conocemos?

$\hat{Y} = \max\{P(Y = l), P(Y = p)\}$



- En una caja se tienen lápices y plumas de color azul y negro (ningún otro particular las diferencia, es decir, tiene misma forma, etc.)

	Azul	Negro
Lápiz	7	3
Pluma	1	4

- Con base a la información que conocemos, podemos sacar las siguientes probabilidades (a priori):

$$p(Y = l) = {}^{2}/_{3}$$

$$p(X = a|Y = l) = {}^{7}/_{10}$$

$$p(X = n|Y = l) = {}^{3}/_{10}$$

$$p(Y = p) = {}^{1}/_{3}$$

$$p(X = a|Y = p) = {}^{1}/_{5}$$

$$p(X = n|Y = p) = {}^{4}/_{5}$$

- Experimento: se saca un elemento al azar de la caja, con solo observar su color… definir si es una pluma o un lápiz; es decir:

$$\left.\begin{array}{r} p(Y = p|X = c) \\ p(Y = l|X = c) \end{array}\right\} \text{Probabilidad a posteriori}$$

- Solución: ¡Teorema de Bayes!

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} = \frac{p(X|Y)p(Y)}{\sum_Y p(X|Y)p(Y)}$$

- Ejemplo: Calcular la probabilidad de que al extraer un elemento color azul, este sea un lápiz:

Probabilidad Marginal

$p(Y = l) = {}^{2}/_{3}$

$p(Y = p) = {}^{1}/_{3}$

Probabilidad Condicional

$p(X = a|Y = l) = {}^{7}/_{10}$

$p(X = n|Y = l) = {}^{3}/_{10}$

$p(X = a|Y = p) = {}^{1}/_{5}$

$p(X = n|Y = p) = {}^{4}/_{5}$

$$p(Y = l|X = a) = \frac{p(X = a|Y = l)p(Y = l)}{\sum_{Y} p(X = a|Y)p(Y)}$$

$$p(Y = l|X = a) = \frac{({}^{7}/_{10})({}^{2}/_{3})}{({}^{7}/_{10})({}^{2}/_{3}) + ({}^{1}/_{5})({}^{1}/_{3})} = \frac{7}{8}$$

- Ejercicio:

Calcular la probabilidad de que al extraer un elemento color negro, este sea una pluma:

Probabilidad Marginal

$$p(Y = l) = {}^{2}/_{3}$$

$$p(Y = p) = {}^{1}/_{3}$$

Probabilidad Condicional

$$p(X = a|Y = l) = {}^{7}/_{10}$$

$$p(X = n|Y = l) = {}^{3}/_{10}$$

$$p(X = a|Y = p) = {}^{1}/_{5}$$

$$p(X = n|Y = p) = {}^{4}/_{5}$$

$$p(Y = p|X = n)$$

- El Clasificador Bayes Ingenuo (Naive Bayes), es el Bayesiano más sencillo, pero ha tenido gran desempeño.

$$\hat{Y} = \max_{C_k} p(Y = C_k | X)$$

Clasificación (Predicción)

Etiqueta de clase

Característica de patrón

- Extendiendo el Teorema de Bayes a muchas características:
  - $X \in \mathbb{R}^d$

$$p(Y = C_k | X = x) = \frac{p(X = x | Y = C_k) p(Y = C_k)}{p(X = x)}$$

$$p(Y = C_k | X = x) = \frac{p(X_1 = x_1, X_2 = x_2, \dots, X_d = x_d | Y = C_k) p(Y = C_k)}{p(X_1 = x_1, X_2 = x_2, \dots, X_d = x_d)}$$

– Asumiendo independencia entre variables del vector de características en:

$$p(Y = C_k | X = x) = p(Y = C_k | X_1 = x_1, \dots, X_d = x_d)$$

Entonces, el clasificador Bayes Ingenuo es:

$$\hat{Y} = \max_{C_k} \frac{p(Y = C_k) \prod_j p(X_j | Y = C_k)}{\sum_i p(Y = C_k) \prod_j p(X_j | Y = C_k)}$$

$\mathrm{i} = \{1, \dots, C_k\}$ y $j = \{1, \dots, d\}$

o bien,

$$\hat{Y} = \max_{C_k} p(Y = C_k) \prod_j p(X_j | Y = C_k)$$

- ¿Cómo determino la probabilidad a priori condicional $p(X_j|Y = C_k)$?
  - Solución: Asumiendo una probabilidad previamente. La más común y usada es la Distribución Normal.

$$p(X_j|Y = C_{kj}) = \frac{1}{\sqrt{2\pi\sigma_{kj}}} e^{\left(-\frac{1}{2}\left[\frac{x-\mu_{kj}}{\sigma_{kj}}\right]^2\right)}$$

- μ_{kj} es la media de la j-ésima característica en la k-ésima clase.
- σ_{kj} es la desviación estándar de la j-ésima característica en la k-ésima clase.

- Dado un conjunto de entrenamiento $\{(x_i, y_i)\}$ del dataset iris plant (conformado con los primeros 45 patrones de cada clase y usando 3ra y 4ta características), se procede a calcular la media y desviación estándar de cada característica por clase:

	C3 - μ/σ	C4 - μ/σ
Setosa	1.46/0.18	0.25/0.11
Versicolor	4.29/0.45	1.34/0.20
Virginica	5.59/0.56	2.02/0.28

- Para clasificar un vector $X = [5.1, 1.8]$, se procede de la siguiente forma:

$$\hat{Y} = \max \begin{cases} p(Y=0)\prod_{j=1}^{2} p(X_j|Y=0) \\ p(Y=1)\prod_{j=1}^{2} p(X_j|Y=1) \\ p(Y=2)\prod_{j=1}^{2} p(X_j|Y=2) \end{cases}$$

$$p(Y=0)\prod_{j=1}^{2} p\left(X_j \middle| Y=0\right) =$$

$$p(Y=0)[p(X_1=5.1|Y=0)p(X_2=1.8|Y=0)] =$$

$$\left(\frac{45}{135}\right)\left\{\left[\frac{1}{\sqrt{2\pi 0.18}}e^{\left(-\frac{1}{2}\left[\frac{5.1-1.46}{0.18}\right]^2\right)}\right]\left[\frac{1}{\sqrt{2\pi 0.11}}e^{\left(-\frac{1}{2}\left[\frac{1.8-0.25}{0.11}\right]^2\right)}\right]\right\}=$$

4.5831E-133

$$p(Y=1)\prod_{j=1}^{2} p\left(X_j \middle| Y=1\right) =$$

$$p(Y=1)[p(X_1=5.1|Y=1)p(X_2=1.8|Y=1)] =$$

$$\left(\frac{45}{135}\right)\left\{\left[\frac{1}{\sqrt{2\pi 0.45}}e^{\left(-\frac{1}{2}\left[\frac{5.1-4.29}{0.45}\right]^2\right)}\right]\left[\frac{1}{\sqrt{2\pi 0.20}}e^{\left(-\frac{1}{2}\left[\frac{1.8-1.34}{0.20}\right]^2\right)}\right]\right\} =$$

0.00248492

$$p(Y = 2)\prod_{j=1}^{2} p\left(X_j \middle| Y = 2\right) =$$

$$p(Y = 2)[p(X_1 = 5.1|Y = 2)p(X_2 = 1.8|Y = 2)] =$$

$$\left(\frac{45}{135}\right)\left\{\left[\frac{1}{\sqrt{2\pi 0.56}}e^{\left(-\frac{1}{2}\left[\frac{5.1-5.59}{0.56}\right]^2\right)}\right]\left[\frac{1}{\sqrt{2\pi 0.28}}e^{\left(-\frac{1}{2}\left[\frac{1.8-0.2.02}{0.28}\right]^2\right)}\right]\right\}=$$

0.06709919

Clasificador Bayes Ingenuo: Ejemplo

$$\hat{Y} = \max \begin{cases} p(Y = 0) \prod_{j=1}^{2} p\left(X_j \middle| Y = 0\right) = 4.5831\, E - 133 \\ p(Y = 1) \prod_{j=1}^{2} p\left(X_j \middle| Y = 1\right) = 0.00248492 \\ p(Y = 2) \prod_{j=1}^{2} p\left(X_j \middle| Y = 2\right) = 0.06709919 \end{cases}$$

El patrón X es clasificado en la clase con índice 2, la cual en el dataset original es su clase verdadera (Virginica). Las probabilidades obtenidas son:

$p(Y = 0|X = x) =$ 6.5864E-132

$p(Y = 1|X = x) =$ 0.03571096

$p(Y = 2|X = x) =$ 0.96428904

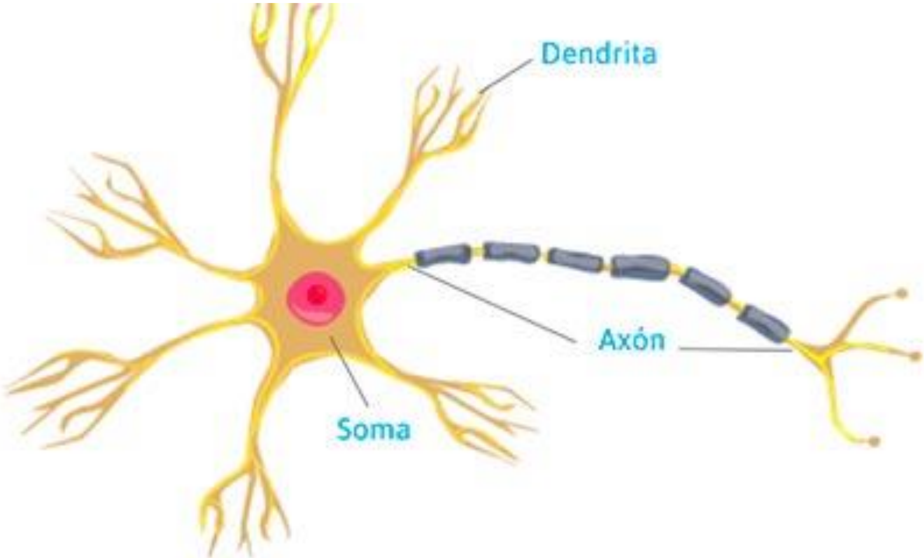
- Dadas las siguientes medias y desviaciones estándar:

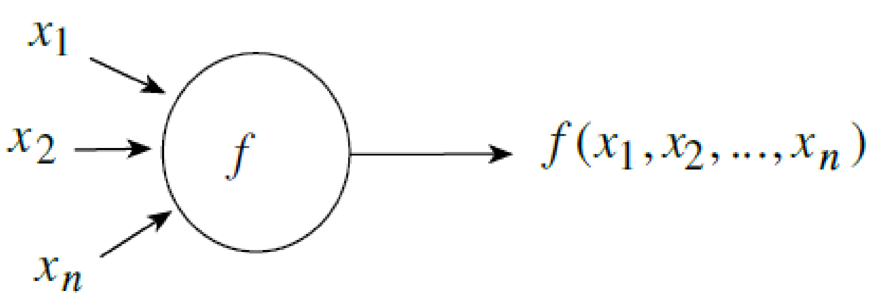
	C3 - μ/σ	C4 - μ/σ
Setosa	1.46/0.18	0.25/0.11
Versicolor	4.29/0.45	1.34/0.20
Virginica	5.59/0.56	2.02/0.28

- Clasificar los siguientes patrones usando el clasificador Bayes ingenuo:
  - $u_1 = (1.4, 0.4)$
  - $u_2 = (5.1, 1.6)$
  - $u_3 = (5.2, 2.3)$
- Obtenga la exactitud.

Métodos de Neuronales

- Neurona de McCulloch-Pitts
- Perceptron
- Perceptron Paralelo



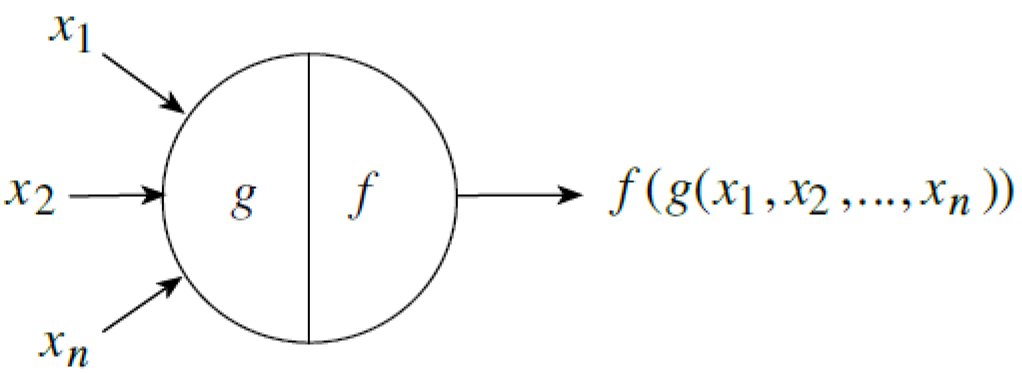


Neurona biológica:

- Dendritas – recolectan señales de otras neuronas
- Soma – procesan la información
- Axón – medio por el cual se propaga la señal de salida.

Neurona artificial:

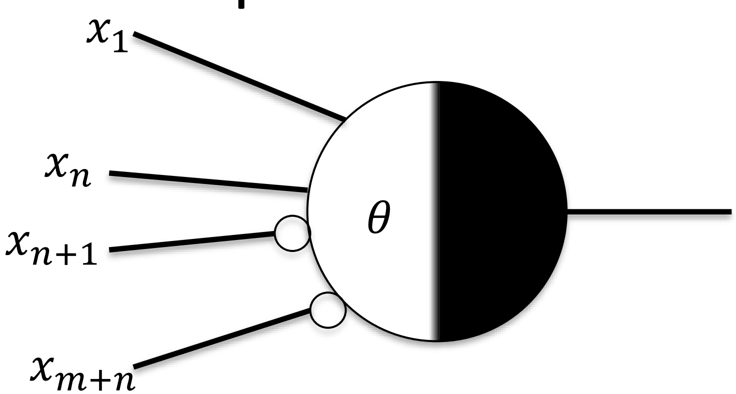
- Entradas – x vector n-dimensional
- f – función "primitiva"
- $f(x_1, \ldots, x_n)$ – respuesta

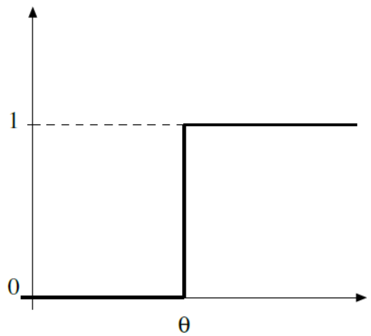


Neurona artificial genérica:

- $g(\cdot)$ – función de integración
- $f(\cdot)$ – función de activación (o salida)

- Características:
  - Acepta solo señales binarias.
  - Produce salidas binarias.
  - Conexiones sinápticas
    - Transmiten puros 0's o 1's.
    - Son dirigidas, sin pesos sinápticos y pueden ser excitatorias o inhibitorias.
  - Está provista de un valor de umbral θ.





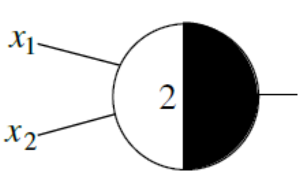
- Funcionamiento:
  - Asuma que a una neurona McCulloch-Pitts llega una entrada $x_1, \dots, x_n$ a través de n conexiones excitatorias y una entrada $y_1, \dots, y_m$ mediante m conexiones inhibitorias.
  - Si $m \geq 1$ y al menos una de las señales $y_1, \dots, y_m$ es 1, la neurona es inhibida y su salida es 0.
  - De lo contrario, se calcula la excitación total ($x = x_1 + \cdots + x_n$) y compara con el umbral θ de la neurona (si $n = 0$ entonces $x = 0$). Si $x \geq \theta$ la neurona *dispara* un 1. Si $x < \theta$ la salida es 0.

Neurona de McCulloch-Pitts: Síntesis de funciones lógicas

Las funciones lógicas, pueden ser consideradas como un problema de clasificación si definimos las salidas $\{0,1\}$ como las etiquetas de clase, y estás a su vez pueden ser resueltas por la neurona de McCulloch-Pitts.

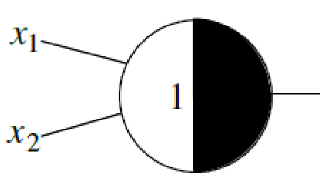
x_1	x_2	Y
0	0	0
0	1	0
1	0	0
1	1	1

AND



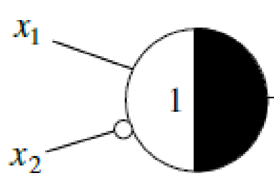
x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	1

OR



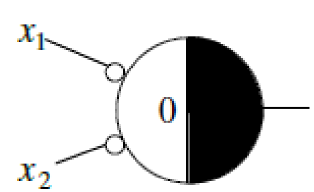
x_1	x_2	Y
0	0	0
0	1	0
1	0	1
1	1	0

x_1 AND $\neg x_2$



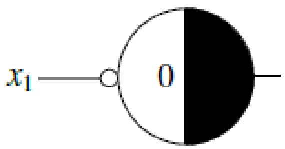
x_1	x_2	Y
0	0	1
0	1	0
1	0	0
1	1	0

NOR

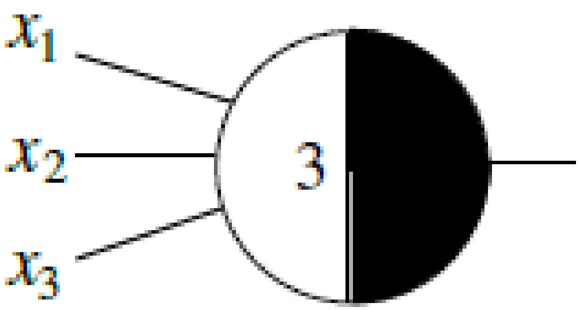


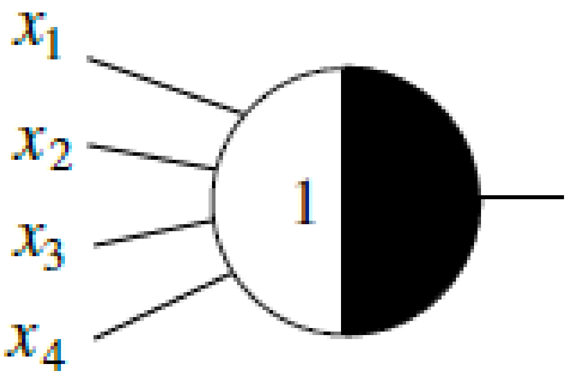
x_1	Y
0	1
1	0

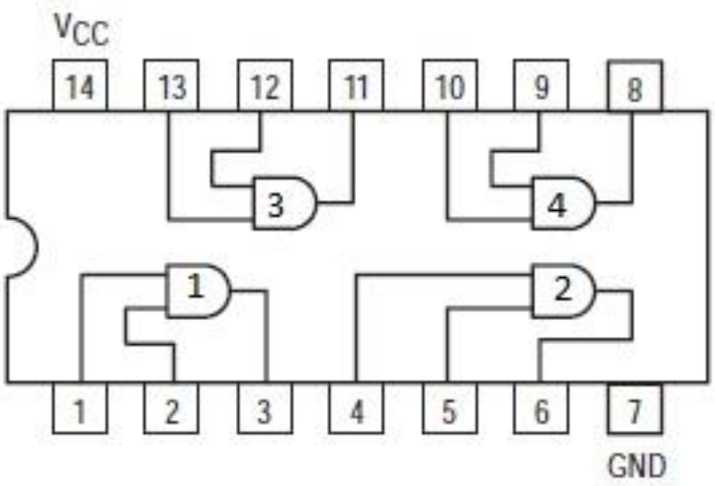
NOT

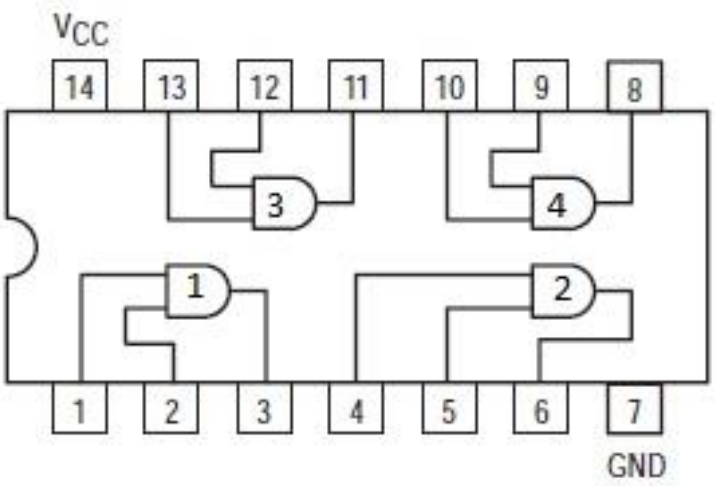


Neurona de McCulloch-Pitts vs Compuertas Lógicas

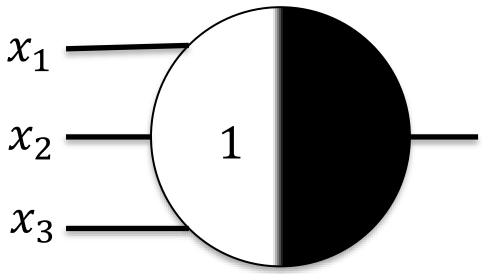


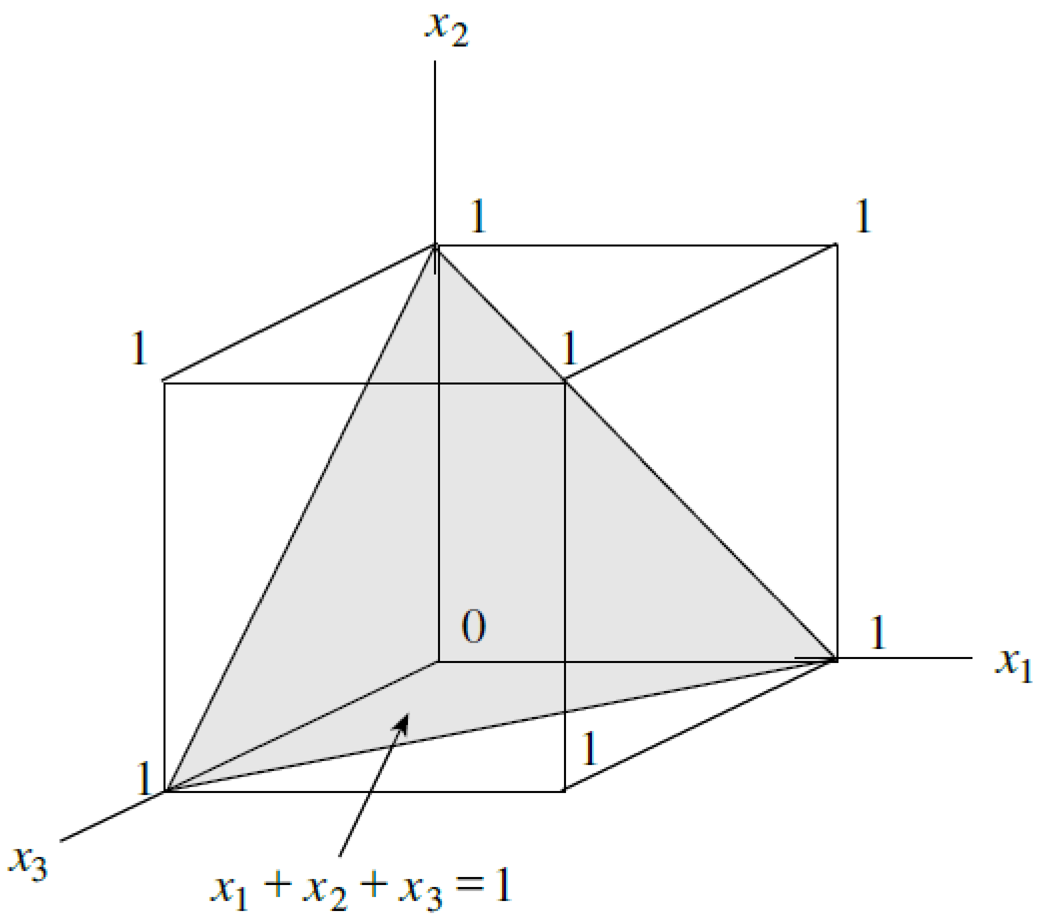






- Una neurona de McCulloch-Pitts divide un espacio de características en dos.





- Encontrar una solución a partir de la neurona de McCulloch-Pitts para la función lógica dada por la siguiente tabla de verdad:

x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Neurona de McCulloch-Pitts: Ejercicio - Solución

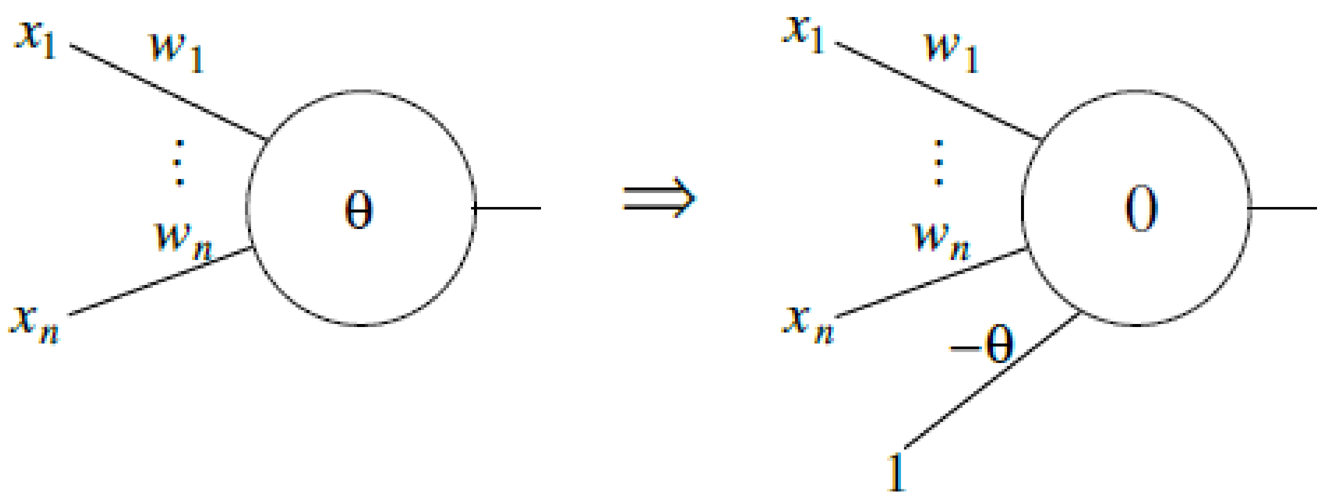
- La función lógica del ejemplo es una XOR, esta se puede resolver a partir de la expresión:
$x_1 XOR x_2 \equiv [x_1 AND \neg x_2] OR [x_2 AND \neg x_1]$

$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	**Y**
0	0	0
0	1	1
1	0	1
1	1	0

$\boldsymbol{x_1}$	$\boldsymbol{\neg x_2}$	$x_3 = x_1 AND \neg \boldsymbol{x_2}$	$\boldsymbol{\neg x_1}$	$\boldsymbol{x_2}$	$x_4 = x_1 AND \neg \boldsymbol{x_2}$	$\boldsymbol{x_3 OR x_4}$
0	1	0	1	0	0	0
0	0	0	1	1	1	1
1	1	1	0	0	0	1
1	0	0	0	1	0	0

- Realizar el diagrama de la función lógica usando neuronas de McCulloch-Pitts

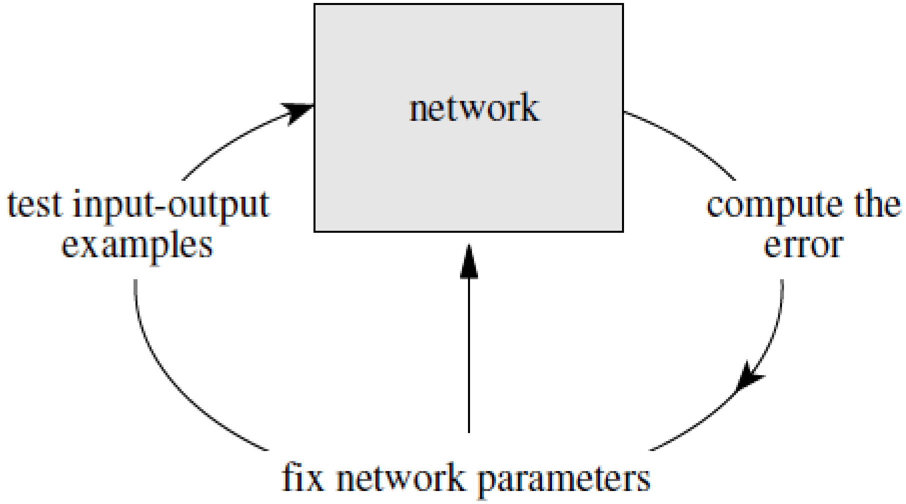
- Es un modelo computacional propuesto por Frank Rosenblatt en 1958, el cual es más general que la neurona de McCulloch-Pitts.



$$f(x,w) = \begin{cases} 1\ si\ \sum_{i=1}^{n} x_i w_i \geq \theta \\ 0\ de\ lo\ contrario \end{cases}$$

$$f(x,w) = \begin{cases} 1\ si\ -\theta + \sum_{i=1}^{n} x_i w_i \geq 0 \\ 0\ de\ lo\ contrario \end{cases}$$

- Un algoritmo de aprendizaje es un método adaptivo por el cual una red de neuronas (o unidades de computo) se auto organiza para implementar el comportamiento deseado.



- Dado un conjunto de entrenamiento de dos conjuntos P y N e un espacio de características n-dimensional (extendido). El algoritmo de aprendizaje del perceptrón es:
  - **start:** El vector de pesos w_0 es generado aleatoriamente en $t = 0$
  - **test:** Un vector $x \in P \cup N$ es seleccionado aleatoriamente:
    - Si $x \in P$ y $w_t \cdot x > 0$ ir a test,
    - Si $x \in P$ y $w_t \cdot x \leq 0$ ir a add,
    - Si $x \in N$ y $w_t \cdot x < 0$ ir a test,
    - Si $x \in N$ y $w_t \cdot x \geq 0$ ir a subtract.
  - **add:** $\boldsymbol{w_{t+1} = w_t + x}$ y $\boldsymbol{t = t + 1}$ ir a test
  - **subtract:** $\boldsymbol{w_{t+1} = w_t - x}$ y $\boldsymbol{t = t + 1}$ ir a test

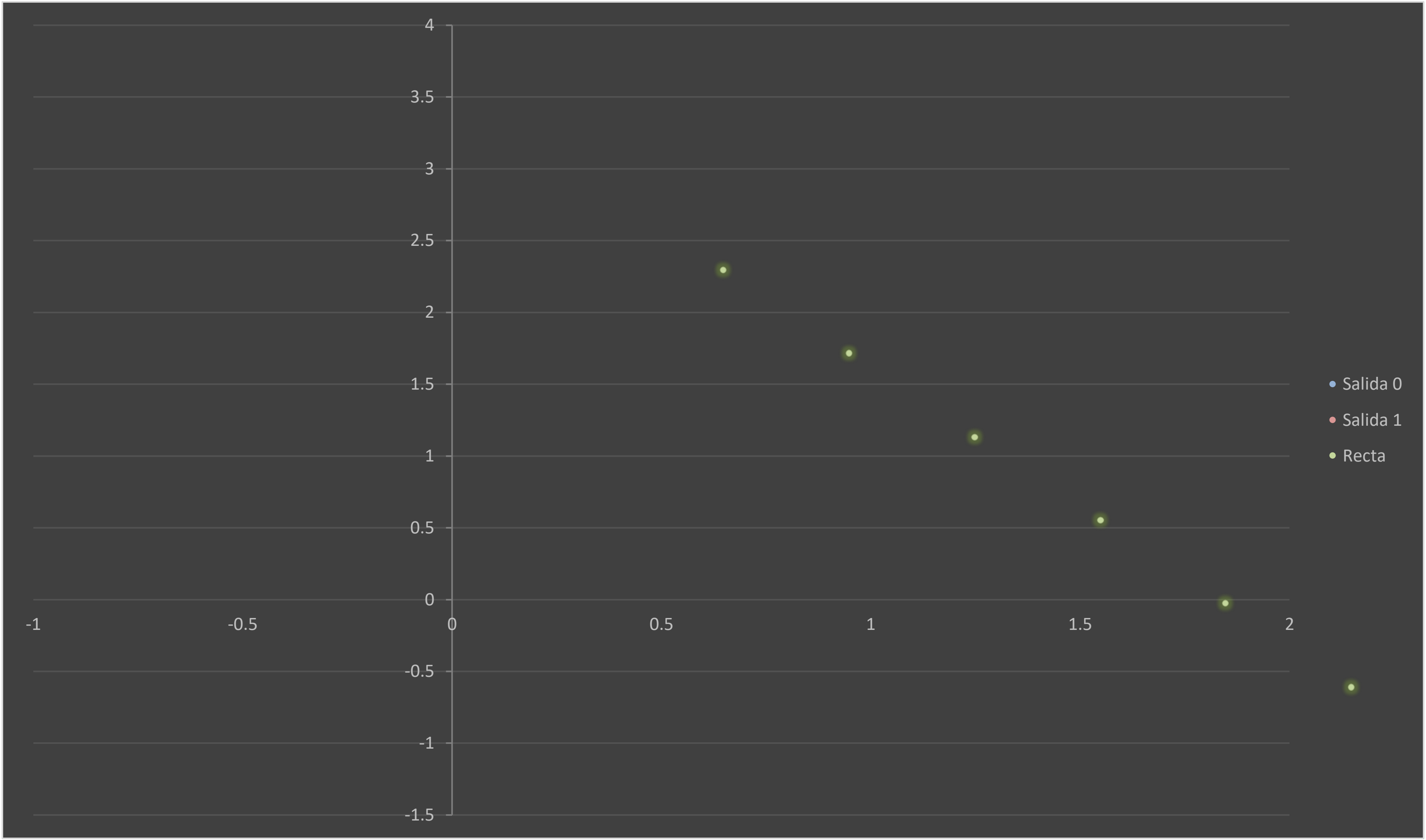
- El Teorema de Convergencia del Perceptron garantiza que si dos conjuntos P y N son linealmente separables, el vector w es actualizado en un numero finito de veces.
- El algoritmo puede ser detenido cuando todos los vectores son correctamente clasificados.

El Perceptron: Solución AND


```
Percepron - Problema AND
w0 = [ 0.34306229 -0.30575336  0.2146545 ]
Epoca: 1
x = [0 0 1], y=0, wo = [ 0.34306229 -0.30575336  0.2146545 ], p = 1, wn = [ 0.34306229 -0.30575336 -0.7853455 ]
x = [0 1 1], y=0, wo = [ 0.34306229 -0.30575336 -0.7853455 ], p = 0, wn = [ 0.34306229 -0.30575336 -0.7853455 ]
x = [1 0 1], y=0, wo = [ 0.34306229 -0.30575336 -0.7853455 ], p = 0, wn = [ 0.34306229 -0.30575336 -0.7853455 ]
x = [1 1 1], y=1, wo = [ 0.34306229 -0.30575336 -0.7853455 ], p = 0, wn = [1.34306229 0.69424664 0.2146545 ]
Epoca: 2
x = [0 0 1], y=0, wo = [1.34306229 0.69424664 0.2146545 ], p = 1, wn = [ 1.34306229  0.69424664 -0.7853455 ]
x = [0 1 1], y=0, wo = [ 1.34306229  0.69424664 -0.7853455 ], p = 0, wn = [ 1.34306229  0.69424664 -0.7853455 ]
x = [1 0 1], y=0, wo = [ 1.34306229  0.69424664 -0.7853455 ], p = 1, wn = [ 0.34306229  0.69424664 -1.7853455 ]
x = [1 1 1], y=1, wo = [ 0.34306229  0.69424664 -1.7853455 ], p = 0, wn = [ 1.34306229  1.69424664 -0.7853455 ]
Epoca: 3
x = [0 0 1], y=0, wo = [ 1.34306229  1.69424664 -0.7853455 ], p = 0, wn = [ 1.34306229  1.69424664 -0.7853455 ]
x = [0 1 1], y=0, wo = [ 1.34306229  1.69424664 -0.7853455 ], p = 1, wn = [ 1.34306229  0.69424664 -1.7853455 ]
x = [1 0 1], y=0, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 0, wn = [ 1.34306229  0.69424664 -1.7853455 ]
x = [1 1 1], y=1, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 1, wn = [ 1.34306229  0.69424664 -1.7853455 ]
Epoca: 4
x = [0 0 1], y=0, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 0, wn = [ 1.34306229  0.69424664 -1.7853455 ]
x = [0 1 1], y=0, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 0, wn = [ 1.34306229  0.69424664 -1.7853455 ]
x = [1 0 1], y=0, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 0, wn = [ 1.34306229  0.69424664 -1.7853455 ]
x = [1 1 1], y=1, wo = [ 1.34306229  0.69424664 -1.7853455 ], p = 1, wn = [ 1.34306229  0.69424664 -1.7853455 ]
Converge
Total de cambios: 6 - W: [ 1.34306229  0.69424664 -1.7853455 ]

Validacion
x = [0 0 1], y=0, p = 0
x = [0 1 1], y=0, p = 0
x = [1 0 1], y=0, p = 0
x = [1 1 1], y=1, p = 1
```

El Perceptron: Solución Geométrica AND

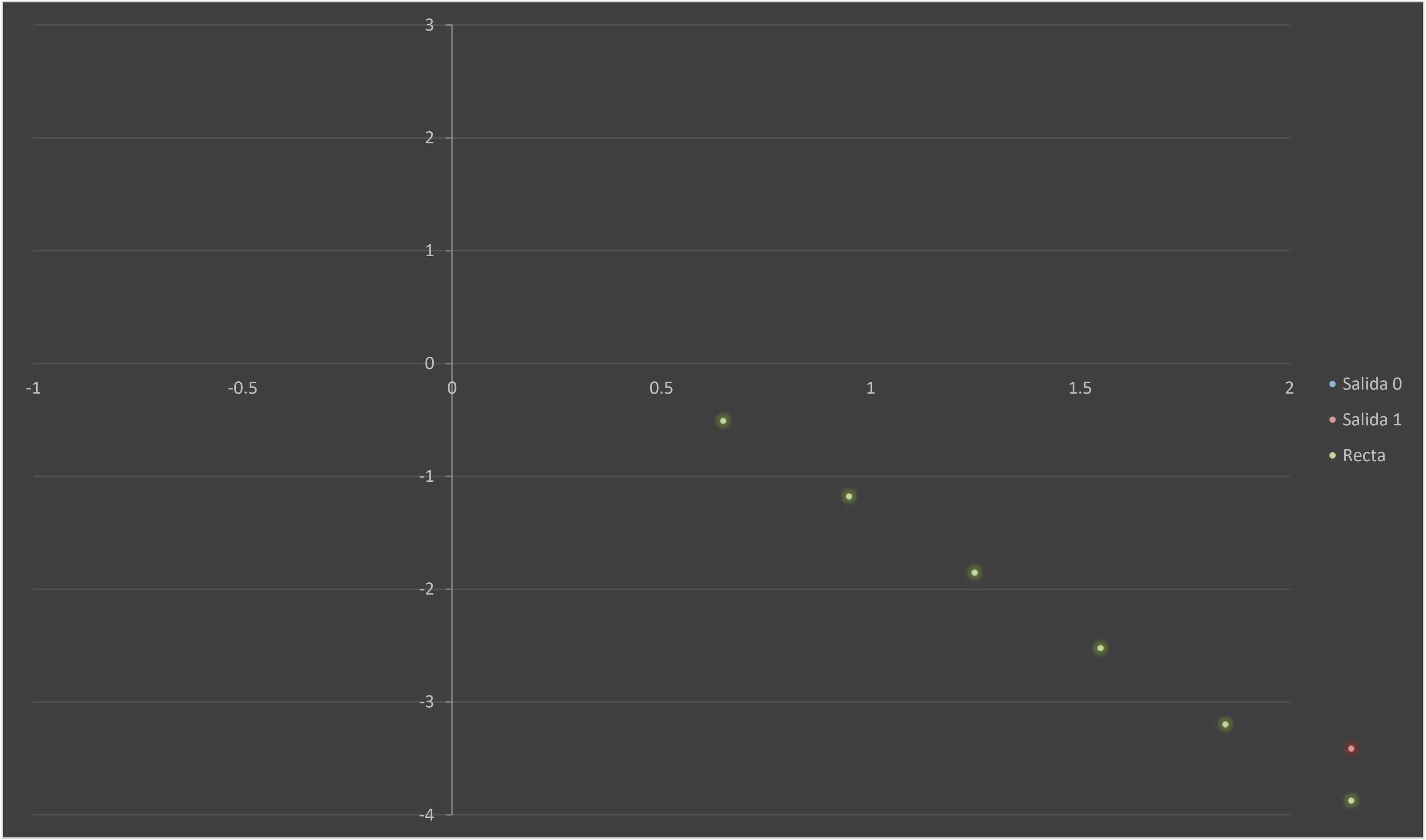


El Perceptron: Solución OR

```
Percepron - Problema OR
w0 = [-0.44459605 -0.3067373   0.41248197]
Epoca: 1
x = [0 0 1], y=0, wo = [-0.44459605 -0.3067373   0.41248197], p = 1, wn = [-0.44459605 -0.3067373  -0.58751803]
x = [0 1 1], y=1, wo = [-0.44459605 -0.3067373  -0.58751803], p = 0, wn = [-0.44459605  0.6932627   0.41248197]
x = [1 0 1], y=1, wo = [-0.44459605  0.6932627   0.41248197], p = 0, wn = [0.55540395 0.6932627  1.41248197]
x = [1 1 1], y=1, wo = [0.55540395 0.6932627  1.41248197], p = 1, wn = [0.55540395 0.6932627  1.41248197]
Epoca: 2
x = [0 0 1], y=0, wo = [0.55540395 0.6932627  1.41248197], p = 1, wn = [0.55540395 0.6932627  0.41248197]
x = [0 1 1], y=1, wo = [0.55540395 0.6932627  0.41248197], p = 1, wn = [0.55540395 0.6932627  0.41248197]
x = [1 0 1], y=1, wo = [0.55540395 0.6932627  0.41248197], p = 1, wn = [0.55540395 0.6932627  0.41248197]
x = [1 1 1], y=1, wo = [0.55540395 0.6932627  0.41248197], p = 1, wn = [0.55540395 0.6932627  0.41248197]
Epoca: 3
x = [0 0 1], y=0, wo = [0.55540395 0.6932627  0.41248197], p = 1, wn = [ 0.55540395  0.6932627  -0.58751803]
x = [0 1 1], y=1, wo = [ 0.55540395  0.6932627  -0.58751803], p = 1, wn = [ 0.55540395  0.6932627  -0.58751803]
x = [1 0 1], y=1, wo = [ 0.55540395  0.6932627  -0.58751803], p = 0, wn = [1.55540395 0.6932627  0.41248197]
x = [1 1 1], y=1, wo = [1.55540395 0.6932627  0.41248197], p = 1, wn = [1.55540395 0.6932627  0.41248197]
Epoca: 4
x = [0 0 1], y=0, wo = [1.55540395 0.6932627  0.41248197], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [0 1 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [1 0 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [1 1 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
Epoca: 5
x = [0 0 1], y=0, wo = [ 1.55540395  0.6932627  -0.58751803], p = 0, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [0 1 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [1 0 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
x = [1 1 1], y=1, wo = [ 1.55540395  0.6932627  -0.58751803], p = 1, wn = [ 1.55540395  0.6932627  -0.58751803]
Converge
Total de cambios: 7 - W: [ 1.55540395  0.6932627  -0.58751803]

Validacion
x = [0 0 1], y=0, p = 0
x = [0 1 1], y=1, p = 1
x = [1 0 1], y=1, p = 1
x = [1 1 1], y=1, p = 1
```

El Perceptron: Solución Geométrica OR



- Con el algoritmo de aprendizaje original del Perceptron, si el conjunto de datos no es linealmente separable el algoritmo no termina.
- El Algoritmo del bolsillo permite calcular la separación lineal que clasifica correctamente la mayor cantidad de vectores en el conjunto positivo P y en el conjunto negativo N.

- Dado un conjunto de entrenamiento de dos conjuntos P y N e un espacio de características n-dimensional (extendido). El algoritmo de aprendizaje del bolsillo es:
  - **start:** Inicializa el vector de pesos w aleatoriamente. Define un vector de pesos "stored" $w_s = w$. Hacer $h_s = 0$, la historia de w_s
  - **Iterate:** Actualizar w usando una sola iteración del algoritmo de aprendizaje del perceptrón. Mantener registro de la cantidad h de vectores probados exitosamente. Si $h > h_s$, substituir w_s con w y h_s con h. Continuar iterando.

- El perceptrón, pese a su éxito temprano, fue abandonado por su poder expresivo limitado.
- Esta limitante, fue sobrellevada por redes de al menos dos capas de perceptrones con funciones de activación como la sigmoide; a estas redes se les conoce como Perceptrones Multicapa (MLPs) y suelen ser entrenadas con la conocida regla de aprendizaje de Retropropagación (BP).

- El MLP entrenado con la BP implica varios problemas para implementaciones en hardware; e incluso para software, ya que la BP requiere de una función de activación con derivada definida, entre otros inconvenientes.
- Una propuesta "más simple" e implementable fácilmente en hardware y software es: el perceptrón paralelo.

The results for the empirical comparison show the average accuracy on the test set for 10 times 10-fold CV (MADALINE: $n = 3$, MLP: 3 hidden units, SVM: 2nd degree polynomial kernel) and the corresponding standard error

Dataset	*p*-delta ($n = 3$)	MADA LINE	WEKA MLP + BP	WEKA C4.5	WEKA SVM
BC	96.94% ± 0.20	96.28% ± 0.44	96.50% ± 0.19	95.46% ± 0.53	96.87% ± 0.16
CH	97.25% ± 0.23	97.96% ± 0.18	99.27% ± 0.10	99.40% ± 0.07	99.43% ± 0.08
CR	71.73% ± 0.82	70.51% ± 0.99	73.12% ± 0.76	72.72% ± 0.89	75.45% ± 0.75
DI	73.66% ± 1.03	73.37% ± 1.38	76.77% ± 0.60	73.74% ± 0.79	77.32% ± 0.55
HD	80.02% ± 1.19	78.82% ± 1.25	82.09% ± 1.08	76.25% ± 2.22	80.78% ± 1.19
IO	84.78% ± 1.57	86.52% ± 1.23	89.37% ± 0.80	89.74% ± 0.74	91.20% ± 0.53
SI	95.72% ± 0.21	95.73% ± 0.33	96.23% ± 0.27	98.67% ± 0.21	93.92% ± 0.16
SN	74.04% ± 2.96	78.85% ± 3.16	81.63% ± 1.24	73.32% ± 1.90	84.52% ± 1.08

- Un perceptrón con d entradas calcula la siguiente función f de $\mathbb{R}^d \to \{-1,1\}$:

$$f(z) = \begin{cases} 1 & si\ \alpha \cdot z \geq 0 \\ -1 & de\ otra\ manera \end{cases}$$

- Donde $\alpha \in \mathbb{R}^d$ es el vector de pesos del perceptrón y $\alpha \cdot z$ denota el producto punto. Una de las entradas es el “bias”.

- Un perceptrón paralelo se define como una sola capa que consiste de un numero finito de n percetrones (sin conexiones laterales).
- Sea $f_1, \dots, f_n$ las funciones de $\mathbb{R}^d \to \{-1,1\}$ calculadas por los perceptrones. Para una entrada z la salida del perceptrón paralelo es el valor (caso de clasificación binaria)

$$s(p) = \begin{cases} -1 \; si \; p < 0 \\ +1 \; si \; p \geq 0 \end{cases}$$

$$p = \sum_{i=1}^{n} f_i(z) \in \{-n, \dots, n\}$$

- Sea $(z, o) \in \mathbb{R}^d \times [-1, +1]$ el patrón de entrenamiento actual y $\alpha_1, \ldots, \alpha_n \in \mathbb{R}^d$ los vectores de los pesos de n perceptrones individuales en el perceptrón paralelo. Así, la salida actual del perceptrón paralelo es

$$\hat{o} = s(p)$$

- La Regla p-delta esta dada por:

p-delta **rule**

For all $i = 1, \ldots, n$:

(a)

$$\boldsymbol{\alpha}_i \leftarrow \boldsymbol{\alpha}_i + \eta \begin{cases} (-\mathbf{z}) & \text{if } \hat{o} > o + \varepsilon \text{ and } \boldsymbol{\alpha}_i \cdot \mathbf{z} \geq 0 \\ (+\mathbf{z}) & \text{if } \hat{o} < o - \varepsilon \text{ and } \boldsymbol{\alpha}_i \cdot \mathbf{z} < 0 \\ \mu(+\mathbf{z}) & \text{if } \hat{o} \leq o + \varepsilon \text{ and } 0 \leq \boldsymbol{\alpha}_i \cdot \mathbf{z} < \gamma \\ \mu(-\mathbf{z}) & \text{if } \hat{o} \geq o - \varepsilon \text{ and } -\gamma < \boldsymbol{\alpha}_i \cdot \mathbf{z} < 0 \\ 0 & \text{otherwise} \end{cases}$$

(b)

$$\boldsymbol{\alpha}_i \leftarrow \boldsymbol{\alpha}_i / ||\boldsymbol{\alpha}_i|| \ .$$

- Donde
  - Taza de aprendizaje $\eta = {}^{1}/_{4\sqrt{t}}$ (t época actual)
  - $||\alpha_i||$ es la norma del vector α_i
  - Exactitud deseada (error) ε
  - Margen $\gamma = 0.05$
  - Factor de importancia de un margen "limpio" $\mu = 1$
- Los últimos 3 parámetros tienen mas utilidad en problemas de regresión.

El Perceptron Paralelo: Ejemplo Clasificación

UNIVERSIDAD DE GUANAJUATO

```
Alphas iniciales:
[ 0.191171    0.51533588 -0.06464429 -0.82762835 -0.09346124]
[-0.32198254  0.50569968 -0.60027554  0.09797911 -0.52025421]
[ 0.67498121 -0.22298994 -0.68080103 -0.11552985  0.13356146]
z=[0.73333333 0.625      0.82352941 1.         1.        ], o=-1, op=-1
Caso contrario (5) -sin correcion-
z=[0.33333333 0.5        0.11764706 0.         1.        ], o=1, op=1
Caso contrario (5) -sin correcion-
z=[0.6        0.625      0.17647059 0.125      1.        ], o=1, op=1
Caso contrario (5) -sin correcion-
z=[0.    0.375 0.    0.    1.   ], o=1, op=1
Caso contrario (5) -sin correcion-
z=[0.66666667 1.         0.94117647 1.         1.        ], o=-1, op=-1
Caso contrario (5) -sin correcion-
z=[0.4        0.         0.47058824 0.25       1.        ], o=1, op=-1
Correcion de alpha=[ 0.191171    0.51533588 -0.06464429 -0.82762835 -0.09346124]
Caso (2) op<o-epsilon y prod_punto<0
Correcion de alpha=[-0.32198254  0.50569968 -0.60027554  0.09797911 -0.52025421]
Caso (2) op<o-epsilon y prod_punto<0
Correcion de alpha=[ 0.67498121 -0.22298994 -0.68080103 -0.11552985  0.13356146]
z=[1.    0.625 1.    0.375 1.   ], o=-1, op=-1
Caso contrario (5) -sin correcion-
z=[0.8        0.75       0.47058824 0.25       1.        ], o=1, op=1
Caso contrario (5) -sin correcion-
z=[0.26666667 0.625      0.47058824 0.625      1.        ], o=-1, op=-1
Caso contrario (5) -sin correcion-
z=[0.26666667 0.125      0.88235294 0.125      1.        ], o=-1, op=-1
Caso contrario (5) -sin correcion-
Alphas finales:
[ 0.29669855  0.52511895  0.05400897 -0.77965345  0.15951047]
[-0.27818987  0.63374592 -0.60483295  0.2011134  -0.33868422]
[ 0.67498121 -0.22298994 -0.68080103 -0.11552985  0.13356146]
```

El Perceptron Paralelo: Ejemplo Regresión

UNIVERSIDAD DE GUANAJUATO

ParallelPerceptronR

```
Epoch 100
Real [-0.95104895], Predicted -0.0324
Real [-0.95979021], Predicted -0.6874
Real [-0.88636364], Predicted -0.378
Real [-0.93006993], Predicted -0.034
Real [-0.94755245], Predicted -0.936
Real [-0.19755245], Predicted 0.2676
Real [-0.89685315], Predicted -0.3524
Real [-0.93181818], Predicted -0.4906
Real [-0.96678322], Predicted -0.9418
Real [-0.96328671], Predicted -0.7804
Real [-0.8479021], Predicted -0.7938
Real [-0.47377622], Predicted -0.3886
Real [-0.87937063], Predicted -0.719
Real [-0.81293706], Predicted -0.9996
Real [-0.94405594], Predicted -0.996
Real [-0.90909091], Predicted -1
Real [-0.82692308], Predicted -0.4406
Real [-0.93181818], Predicted -0.9394
Real [-0.96153846], Predicted -0.973
Real [-0.97902098], Predicted -0.997
Real [-0.91958042], Predicted -0.38
Real [-0.96328671], Predicted -0.9908
Real [-0.76398601], Predicted -0.6048
Real [-0.95629371], Predicted -0.9638
Real [-0.77622378], Predicted -0.3912
Real [-0.86013986], Predicted -0.132
Real [1.], Predicted -1
Real [-0.87062937], Predicted -0.3798
Real [-0.63986014], Predicted -0.8982
Real [-0.81118881], Predicted -0.3798
Real [0.98951049], Predicted 0.1666
Real [-0.15559441], Predicted 0.2642
Real [-0.64685315], Predicted -0.7966
Real [-0.94055944], Predicted -0.9994
Real [-0.95979021], Predicted -1
Real [-0.93706294], Predicted -0.9204
```

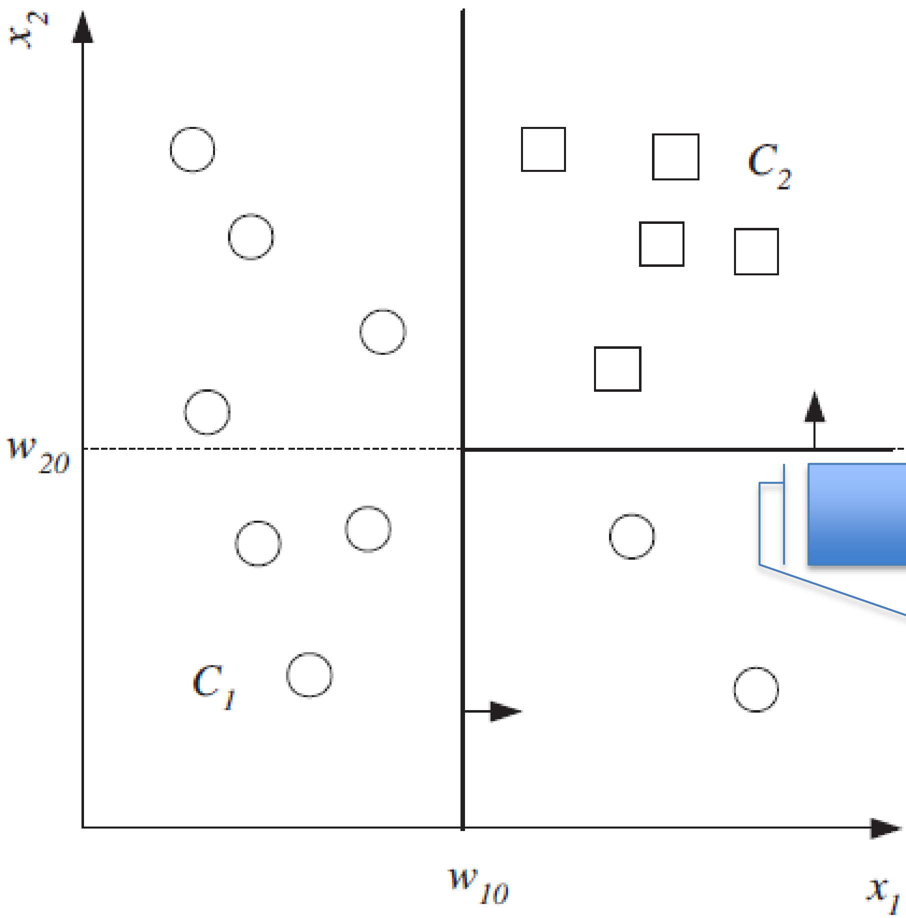
```
Real [-0.93706294], Predicted -0.9204
Real [-0.36888112], Predicted 0.2598
Real [-0.95454545], Predicted -0.033
Real [-0.97377622], Predicted -0.0328
Real [-0.95454545], Predicted -0.3544
Real [-0.98951049], Predicted -0.3784
Real [-0.98951049], Predicted -0.5132
Real [-0.97202797], Predicted -0.9962
Real [-0.98776224], Predicted -0.5152
Real [-0.9458042], Predicted -0.9392
Real [-0.90034965], Predicted -0.7696
Real [-0.95454545], Predicted -1
Real [-0.98251748], Predicted -0.9958
Real [-0.92307692], Predicted -1
Real [-0.9020979], Predicted -0.5178
Real [-0.98251748], Predicted -0.7158
Real [-0.77797203], Predicted -0.804
Real [-0.92307692], Predicted -0.5768
Real [-0.92132867], Predicted -0.6292
Real [-0.30244755], Predicted -0.3642
Real [-0.95454545], Predicted 0.145
Real [-0.9527972], Predicted -0.6038
Real [-0.77972028], Predicted -0.9696
Real [-0.96503497], Predicted -0.965
Real [-0.90559441], Predicted -0.3506
Real [-0.94055944], Predicted -0.7134
Real [-0.97202797], Predicted -0.0328
Real [-0.92482517], Predicted -0.7226
Real [-0.93181818], Predicted -0.8368
Real [-0.98076923], Predicted -1
Real [-0.96328671], Predicted -0.7482
Real [-0.8951049], Predicted -0.3838
Real [-0.90559441], Predicted -0.9068
Real [-0.95454545], Predicted -0.7676
Real [-0.45454545], Predicted -1
Real [-0.81643357], Predicted -0.7836
Real [-0.94405594], Predicted -0.4646
```

```
Real [-0.94405594], Predicted -0.4646
Real [-0.90559441], Predicted -0.9512
Real [-0.98601399], Predicted -0.9954
Real [-0.9527972], Predicted -0.9648
Real [-0.98951049], Predicted -0.0328
Real [-0.84965035], Predicted -0.0878
Real [-0.75874126], Predicted -0.5806
Real [-0.9020979], Predicted -1
Real [-0.77972028], Predicted 0.2086
Real [-0.88636364], Predicted -0.9998
Real [-0.98076923], Predicted -0.9638
Real [-0.98601399], Predicted -0.9776
Real [-0.97552448], Predicted -0.9996
Real [-0.90384615], Predicted -0.7268
Real [-0.63636364], Predicted 0.1786
Real [-0.68706294], Predicted 0.2052
Real [-0.96503497], Predicted -0.3878
Real [-0.97552448], Predicted -0.9972
Real [-0.68006993], Predicted -0.6046
Real [-0.94055944], Predicted -0.3568
Real [-0.95104895], Predicted -0.9938
Real [-0.44055944], Predicted -1
Real [-0.81993007], Predicted -0.7836
Real [-0.95454545], Predicted 0.072
Real [-0.96853147], Predicted -0.9886
Real [-0.83566434], Predicted -0.8948
Real [-0.78321678], Predicted -1
Real [-0.93181818], Predicted -0.9984
Real [-0.11888112], Predicted 0.106
Real [-0.68181818], Predicted -0.9154
Real [-0.62587413], Predicted -1
Real [-0.96853147], Predicted -0.885
Real [-0.99300699], Predicted -0.9976
Real [-0.92307692], Predicted -0.437
Real [-0.36713287], Predicted -0.4226
Real [-0.95804196], Predicted -0.999
Real [-0.95804196], Predicted -0.717
```

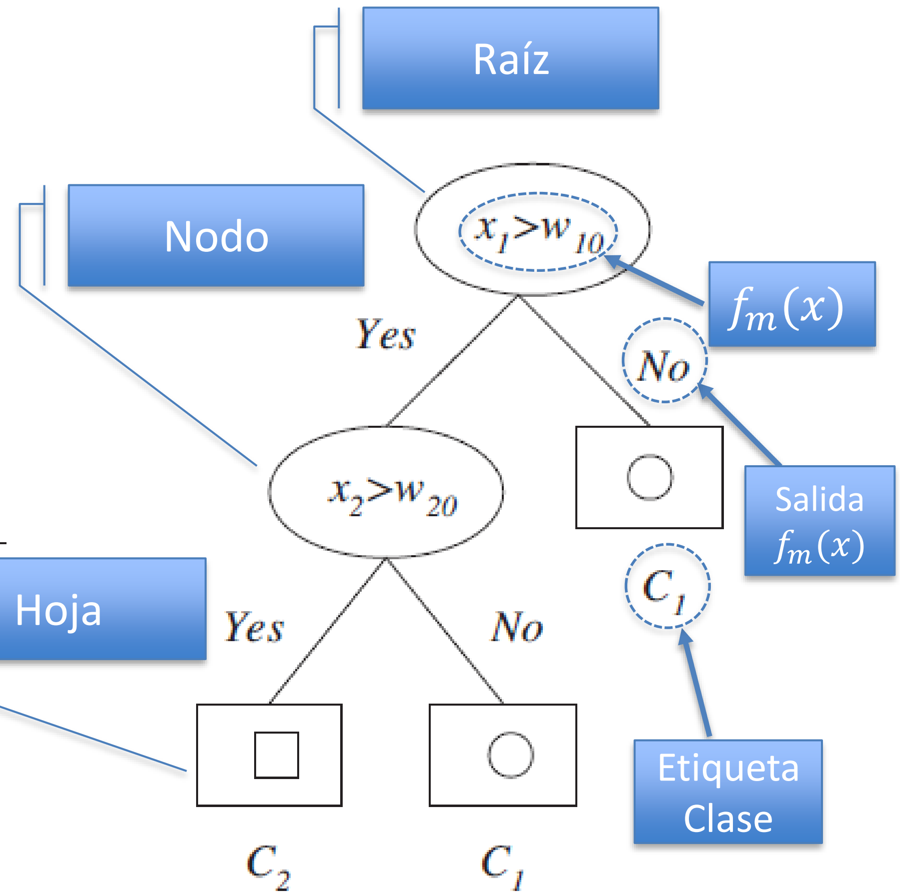
Árboles de Decisión

- Los árboles de decisión son una estructura de datos jerárquica que implementa una estrategia de *divide y vencerás*; estos son ampliamente aceptados debido a la facilidad de interpretación y uso. Es un método donde se divide el espacio de características en regiones locales. Para cada característica se usa el modelo local correspondiente, el cual es calculado para dicha región a partir del conjunto de entrenamiento.

- Un árbol de decisión se compone de: *nodos de decisión internos* y *hojas terminales.*
- Cada nodo de decisión m implementa una función de prueba $f_m(x)$ con resultados discretos que etiquetan las ramas. Dada una entrada, en cada nodo, una prueba es aplicada y una de las ramas es tomada dependiendo de la salida.
- Este proceso inicia en la *raíz* y es repetido recursivamente hasta alcanzar una nodo hoja; el valor escrito en ella es la salida.



Espacio de características



Árbol de Decisión

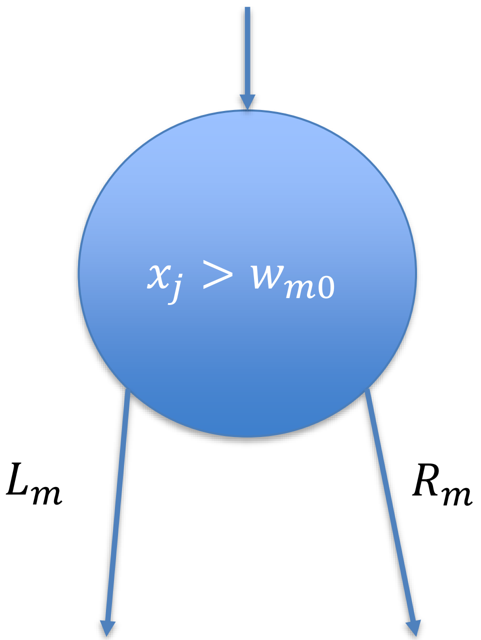
- Los árboles de decisión son modelos no paramétricos debido a que no se asume ninguna forma paramétrica de la densidad de las clases y la estructura del árbol no es definida a priori; el árbol crece, las ramas y hojas son agregadas durante el proceso de aprendizaje.

- Los árboles de decisión sirven para aprendizaje supervisado; tanto clasificación como regresión.
- Este modelo puede ser usado con variables categóricas, **numéricas** o ambas.
- Estos pueden ser convertidos en un conjunto de reglas *IF-THEN.*
- Existen diversos tipos de árboles, dependiendo de la forma en que son generados; esto depende de: $f_m(x)$.

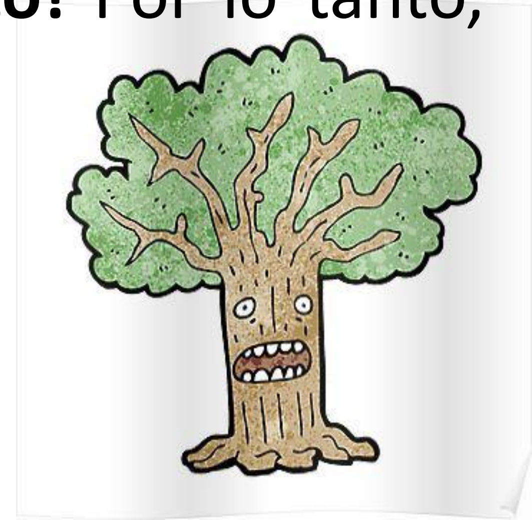
- En un árbol univariado, en cada nodo interno, la prueba usa dolo una de las dimensiones de entrada.
- Los nodos de decisión tiene ramas discretas, entonces, una entrada numérica x_j debe ser discretizada:

$$f_m(x)\colon x_j > w_{m0}$$

- Donde w_{m0} es un valor umbral, lo que permite que el nodo de decisión realice una división binaria:
  - $L_m = \{x | x_j > w_{m0}\}$
  - $R_m = \{x | x_j \leq w_{m0}\}$



- La inducción del árbol es la construcción de este dado conjunto de entrenamiento.
- Dado un conjunto de entrenamiento, existen muchos árboles que lo codifican sin error. Nuestro interés es encontrar el más pequeño entre ellos. El tamaño del árbol es medido como el numero de nodos en el árbol y la complejidad de los árboles de decisión. Encontrar el árbol más pequeño es **¡NP-Completo!** Por lo tanto, hay que usar heurísticas.



- En estos árboles, la buena calidad de una división es cuantificada por una *medida de impuridad*.
- Una división es pura si después de la división, para todas las ramas, todas las instancias que eligen una rama pertenecen a la misma clase.
- Una medida de impuridad es *la entropía*:

$$\beth_m = -\sum_{i=1}^{K} p_m^i \log_2 p_m^i$$

- Donde $0\log0 \equiv 0$ y $p_m^i = {N_m^i}/{N_m}$; N_m es la cantidad de patrones de entrenamiento que llegan al nodo m y N_m^i es la cantidad de patrones que pertenecen a la clase i.

- $\log_b a = {}^{\ln a}/_{\ln b} = \log a / \log b$

- En un problema binario, si $p^1 = 1$ y $p^2 = 0$, todos los ejemplos pertenecen a C^1. En este caso no se requiere seguir dividiendo y la entropía es 0. Si $p^1 = p^2 = 0.5$, la entropía es 1. Entre estos dos valores extremos, se tendrán valores pequeños para la clase más probable y más grandes para la clase menos probable.

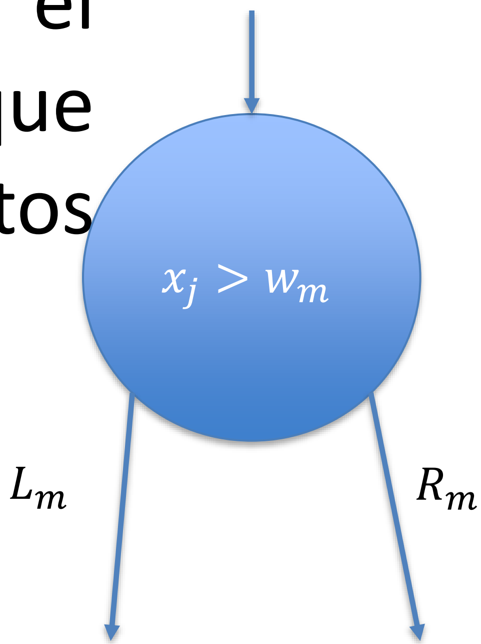
- Ejemplo. En un problema binario, a un nodo m llegan $N_m = 16$ patrones; donde $N_m^1 = 9$ y $N_m^2 = 7$. Calcule la pureza del nodo m con la medida de entropía.
- $\beth_m = -\sum_{i=1}^{2} p_m^i \log_2 p_m^i$
- $\beth_m = -[{}^9/_{16} \log_2 {}^9/_{16} + {}^7/_{16} \log_2 {}^7/_{16}]$
- $\beth_m = -[-0.466917 + (-0.521782)]$
- $\beth_m = 0.988699$

- Ejemplo. En un problema binario, a un nodo m llegan $N_m = 16$ patrones; donde $N_m^1 = 1$ y $N_m^2 = 15$. Calcule la pureza del nodo m con la medida de entropía.
- $\beth_m = -\sum_{i=1}^{2} p_m^i \log_2 p_m^i$
- $\beth_m = -[{}^{1}/_{16} \log_2 {}^{1}/_{16} + {}^{15}/_{16} \log_2 {}^{15}/_{16}]$
- $\beth_m = -[-0.25 + (-0.087290)]$
- $\beth_m = 0.33729$

- Ejemplo. En un problema binario, a un nodo m llegan $N_m = 16$ patrones; donde $N_m^1 = 8$ y $N_m^2 = 8$. Calcule la pureza del nodo m con la medida de entropía.
- $\beth_m = -\sum_{i=1}^{2} p_m^i \log_2 p_m^i$
- $\beth_m = -[{}^{8}/_{16} \log_2 {}^{8}/_{16} + {}^{8}/_{16} \log_2 {}^{8}/_{16}]$
- $\beth_m = -[-0.5 + (-0.5)]$
- $\beth_m = 1$

- Ejemplo. En un problema binario, a un nodo m llegan $N_m = 16$ patrones; donde $N_m^1 = 16$ y $N_m^2 = 0$. Calcule la pureza del nodo m con la medida de entropía.
- $\beth_m = -\sum_{i=1}^{2} p_m^i \log_2 p_m^i$
- $\beth_m = -[{}^{16}/_{16} \log_2 {}^{16}/_{16} + 0]$
- $\beth_m = -[0 + 0]$
- $\beth_m = 0$

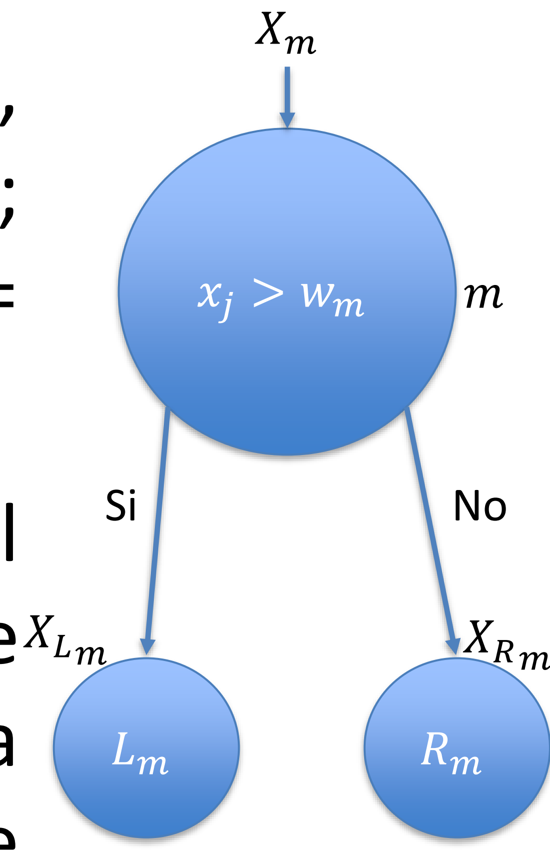
- En cada nodo, se tiene que decidir el conjunto de preguntas candidatas ($f_m(x)$) a ser evaluadas. Cada pregunta corresponde a una división binaria, la cual genera dos nodos descendientes. Esto divide el (sub)conjunto de entrenamiento que alcanza el nodo, en dos subconjuntos disjuntos.



- Debe adoptarse un criterio de división según el cual se elija la mejor división del conjunto de candidatos.
- Se requiere una regla para detener la división que controle el crecimiento del árbol, y entonces un nodo se declara como hoja.
- Se requiere de una regla que asigne a cada hoja una clases específica.

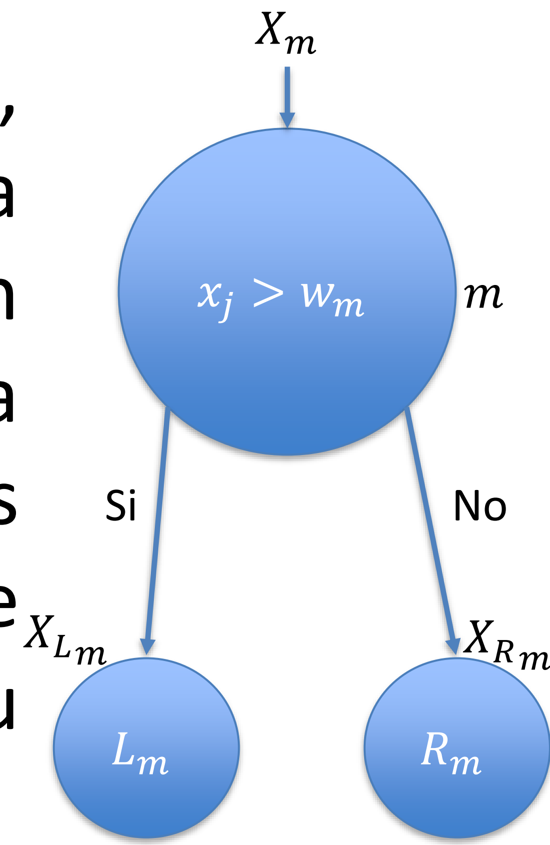
- Las preguntas son de la forma $f_m(x): x_j > w_m$. Para cada característica, cada posible valor del umbral w_m define una división específica del (sub)conjunto de entrenamiento.
- En la práctica, solo se considera un conjunto finito de preguntas. Por ejemplo, para un conjunto X_m con N_m observaciones, cualquiera de sus variables $x_j, j = 1, \dots, d$ puede tomar a lo mucho $N_{mj} \leq N$ diferentes valores. Así, la característica x_j puede usar N_{mj} valores para definir un umbral w_m.

- Cada división binaria de un nodo, genera dos nodos descendientes; $L_m = \{x | x_j > w_{m0}\}$ y $R_m = \{x | x_j \leq w_m\}$.
- La idea para guiar generación del árbol (metodología de crecimiento) es que en cada división genere subconjuntos que son más "clase-homogéneos" comparados con su conjunto antecesor.



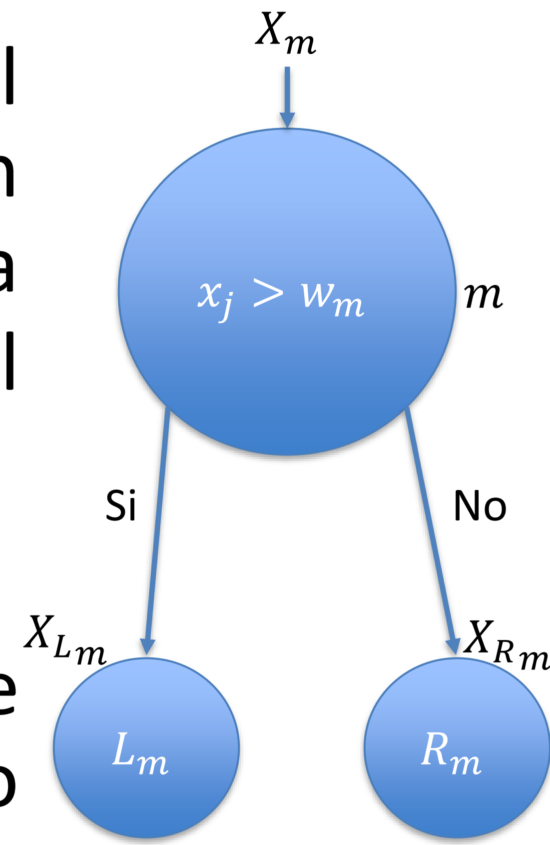
Árboles de Clasificación Univariados: Criterio de División

- La meta es usar una medida, como la entropía, que permita cuantificar la impureza de un nodo y dividir el nodo tal que la impureza general de los nodos descendientes se reduzca de manera óptima con respecto a su ancestro.



Árboles de Clasificación Univariados: Criterio de División

- Al realizar una división, el conjunto de patrones X_m son divididos en $X_{L_m} \cap X_{R_m} = \emptyset$. La disminución de la impureza del nodo se define como:
  - $\Delta\beth_m = \beth_m - {}^{N_{L_m}}/_{N_m}\, \beth_{L_m} - {}^{N_{R_m}}/_{N_m}\, \beth_{R_m}$
- Regresando a la menta, esta se convierte en adoptar del conjunto de preguntas candidatas, aquella que permita la división que tenga el mayor diminución de impurezas.



Árboles de Clasificación Univariados: Regla para detener la división

- Algunas opciones para detener la división son:
  - Integrar un umbral T y detener la división si el máximo valor de $\Delta\beth_m$ de todas las posibles divisiones, es menor que T.
  - Si la cardinalidad del conjunto X que llega al nodo es muy pequeña.
  - Si el conjunto X que llega al nodo es puro, es decir, todos los patrones pertenecen a la misma clase

- Una vez que un nodo es declarado hoja (ya no se divide), una regla común es:

$$c = \arg\max_i N_m^i / N_m$$

- Es decir, la clase se determina de acuerdo a aquella que tenga la mayoría de vectores en el conjunto que llega a la hoja.

1. Comenzar con el nodo raíz, tal que $X_m = X$
2. Por cada nuevo nodo m
    2.1 Para cada característica $x_j, j = 1,2, \ldots, d$
        2.1.1 Para cada valor $w_{\mathrm{i}_{\mathrm{m_j}}}$, $\mathrm{i} = 1,2, \ldots, \mathrm{N_{m_j}}$
            2.1.1.1 Generar L_m y R_m de acuerdo a la pregunta $x_j > w_{i_{m_j}}$
            2.1.1.2 Calcular el decremento de impuridad
        2.1.2 Elegir $w_{\mathrm{i}_{\mathrm{m_0}}}$ que lleve a la mayor diminución de impureza
    2.2 Elegir w_{i_0} que lleve a la mayor diminución de impureza general.
    2.3 Si se llega al criterio para detener la división, declare el nodo m como hoja y asigne su etiqueta de clase
    2.4 Si no, genere los dos nodos descendientes L_m y R_m

UNIVERSIDAD DE GUANAJUATO