# Introducción a Deep Learning con Python

Lic. Celia Cintas – GIBEH CONICET
2 al 4 de Octubre de 2017
Laboratorio de Ciencias de las Imágenes – Universidad Nacional del Sur
Bahía Blanca – Argentina – cad@uns.edu.ar

# Una rápida introducción al aprendizaje de máquina (machine learning).

ML es probablemente la primera en surgir de las múltiples ramas de la Inteligencia Artificial.

Su propósito es obtener sistemas que tengan la capacidad de aprender a realizar alguna tarea o cumplir algún propósito sin requerir una programación explícita.

# Machine Learning: Introducción

Las aplicaciones de ML cubren todo el rango de tareas para las cuales una programación estática sería poco práctica:

- No hay expertos humanos que sepan resolver la tarea.
- Hay expertos humanos, pero éstos no sabrían cómo describir su accionar.
- El contexto es abierto o se modifica.
- Hay muchas instancias (usuarios) de una misma tarea, cada una con restricciones locales diferentes.

# Machine Learning: Introducción

En general, el ML se basa en plantear un «modelo» del dataset (el cual es explícito o implícito, y puede o no ser determinado bajo supervisión).

ML está estrechamente relacionada con los ***analíticos***, en particular con los predictivos (aunque los diagnósticos y los prescriptivos también requieren ML).

También la relación es muy grande con el reconocimiento de patrones y con la minería de datos.

# Machine Learning: Introducción

Otros campos de estudio se enfocan en esta misma problemática:

**Estadística**: Se enfoca en el entendimiento de propiedades (paramétricas) de los fenómenos que generan los datos, con el objetivo de testear diferentes hipótesis acerca de dichos parámetros.

**Data Mining**: Busca patrones, tendencias o relaciones que organicen o simplifiquen los datos con el objeto de hacerlos comprensibles.

**Psicología Cognitiva**: Propone comprender o al menos modelar los mecanismos subyacentes en el aprendizaje de los humanos.

**Teoría de la Ciencia**: Busca conformar teorías descriptivas o normativas acerca del proceso de generación y justificación del conocimiento.

# Tipos de aprendizaje

El aprendizaje, como tarea, se puede clasificar en varias dimensiones.

El aprendizaje **empírico** se apoya en experiencia externa mientras que el **analítico** se basa solo en la descripción del problema.

El aprendizaje **supervisado** incluye casos donde el contexto y el resultado esperado son conocidos, mientras que en el aprendizaje **no supervisado** se desconoce tal resultado.

ML como **simbólico** es más cercano a lo formal/lógico mientras que el **numérico** es cercano al soft computing y al reconocimiento de patrones).

# ML numérico

**Clustering.** Encontrar descripciones compactas que cubra adecuadamente un conjunto de casos con propiedades conocidas.

**Clasificación**. Encontrar un modelo que permita predecir una categoría a partir de casos etiquetados

**Algoritmos genéticos y evolutivos**. Resuelven problemas de optimización con base en una metáfora de la evolución Darwiniana.

**Regresión**. A partir de muestras o mediciones, inducir una función que permita predecir valores o probabilidades.

**Aprendizaje por refuerzo**. Se recompensa o castiga en función del éxito o fracaso en la tarea.

**Detección de atípicos, reducción de dimensionalidad, etc. etc.**

# ML Simbólico

**Aprendizaje “crudo”**. Mapeo uno a uno entre casos y soluciones.

**Aprendizaje basado en casos**. Similar al anterior, pero con algún grado de abstracción.

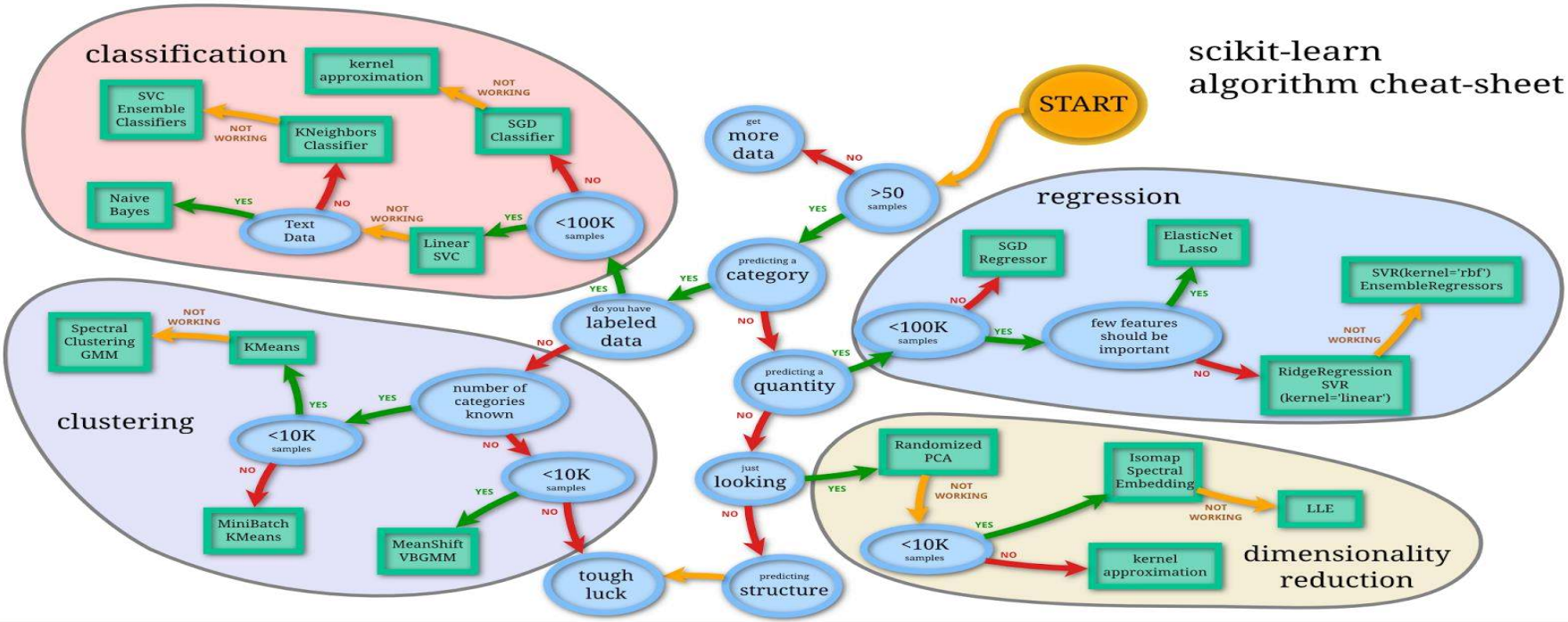
**Inducción**. Partir de casos particulares escogidos, para arribar a alguna generalización.

**Analogía**. Determinar una correspondencia o isomorfismo parcial entredos representaciones.

**Descubrimiento**. Inducción o abducción no supervisada, usualmente en casos donde no se conocen propiedades o valores.

**Aprendizaje basado en explicaciones**. Descubrir conceptos lógicos a partir de ejemplos.

# Roadmap según scikit-learn

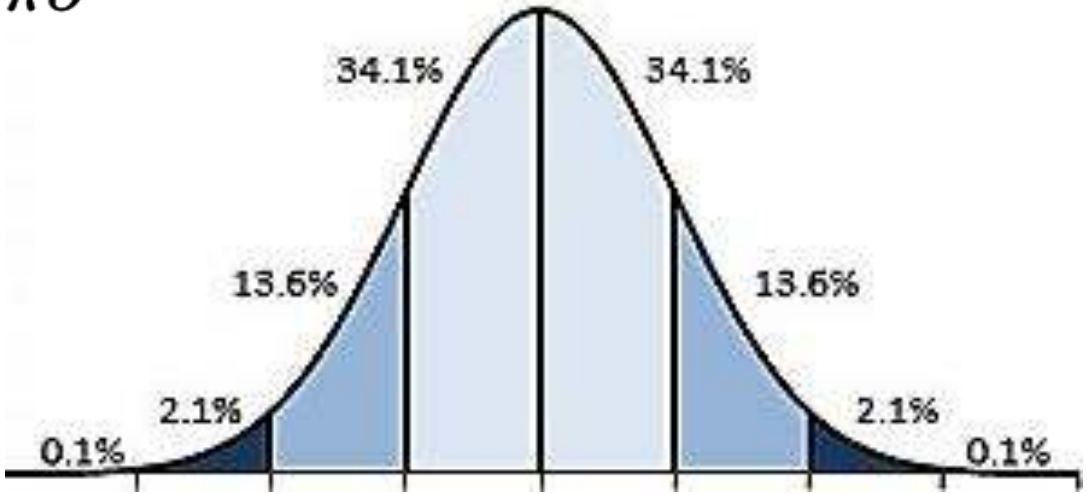


# Estadística

Atributos continuos y discretos, cuantitativos y nominales.
Atributos target y tipos de predictivos.
Espacio de atributos.
Población y muestra. Inferencia estadística.
Estimadores univariados: tendencia central y dispersión.
Distribuciones discretas y contínuas.
Probabilidad como frecuencia relativa.
Distribución normal. Teorema del límite central. Estandarización.
Estimadores multivariados: correlación, covarianza, contingencia.
Intervalo de confianza. Prueba de hipótesis. Significatividad.

# Distribución Normal

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



# Medidas de Evaluación de Modelos

Sensibilidad – Recall
Especificidad
Curva ROC (TPR vs FPR)
Precisión
Exactitud
F-measure
Validación cruzada
RMS
Coeficiente de determinación
Rand/Jacqard

# Medidas de Evaluación de Modelos

| | | predicted condition | | |
|---|---|---|---|---|
| | total population | prediction positive | prediction negative | Prevalence $= \frac{\Sigma \text{ condition positive}}{\Sigma \text{ total population}}$ |
| true condition | condition positive | **True Positive (TP)** | **False Negative (FN)** (type II error) | True Positive Rate (TPR), Sensitivity, Recall, Probability of Detection $= \frac{\Sigma \text{ TP}}{\Sigma \text{ condition positive}}$ |
| | condition negative | **False Positive (FP)** (Type I error) | **True Negative (TN)** | False Positive Rate (FPR), Fall-out, Probability of False Alarm $= \frac{\Sigma \text{ FP}}{\Sigma \text{ condition negative}}$ |
| | Accuracy $= \frac{\Sigma \text{ TP} + \Sigma \text{ TN}}{\Sigma \text{ total population}}$ | Positive Predictive Value (PPV), Precision $= \frac{\Sigma \text{ TP}}{\Sigma \text{ prediction positive}}$ | False Omission Rate (FOR) $= \frac{\Sigma \text{ FN}}{\Sigma \text{ prediction negative}}$ | Positive Likelihood Ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$ |
| | | False Discovery Rate (FDR) $= \frac{\Sigma \text{ FP}}{\Sigma \text{ prediction positive}}$ | Negative Predictive Value (NPV) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ prediction negative}}$ | Negative Likelihood Ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$ |

| | |
|---|---|
| False Negative Rate (FNR), Miss Rate $= \frac{\Sigma \text{ FN}}{\Sigma \text{ condition positive}}$ | |
| True Negative Rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ condition negative}}$ | |
| | Diagnostic Odds Ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$ |

# Roadmap «numérico» más completo

Clasificación
Clustering
Regresión
Reducción de la dimensionalidad
Detección de atípicos
Aprendizaje conjunto
Aprendizaje de estructuras
Minería de texto
Series temporales

# Clasificación

El objetivo es predecir una categoría o clase a partir de otros atributos. Se conoce también como «aprendizaje supervisado», porque se basa en un proceso previo de aprendizaje a partir de ejemplos o casos etiquetados.

Por lo tanto, es necesario conocer a priori los posibles valores de la variable nominal a predecir, y contar con casos de entrenamiento. Un clasificador tiene primero un «contexto de aprendizaje» donde utiliza dichos ejemplos para generar un modelo predictivo.

Cuando los parámetros de evaluación del modelo son satisfactorios, el clasificador se puede poner en producción (etiquetar casos desconocidos).

# Clasificación Estadística Paramétrica

Algunas definiciones previas:

- $p(\omega_i)$ es la probabilidad *a priori* de ocurrencia de un evento de la clase $\omega_i$. Normalmente estas probabilidades se conocen o se pueden estimar.
- $p(\mathbf{x}|\omega_i)$ es la probabilidad condicional, o función de densidad probabilística, (también "verosimilitud") de que se observe un patrón $\mathbf{x}$ cuando el evento es de la clase $\omega_i$.
- $p(\omega_i|\mathbf{x})$ es la probabilidad *a posteriori*, de que el evento sea perteneciente a la clase $\omega_i$ cuando el patrón observado fue $\mathbf{x}$.

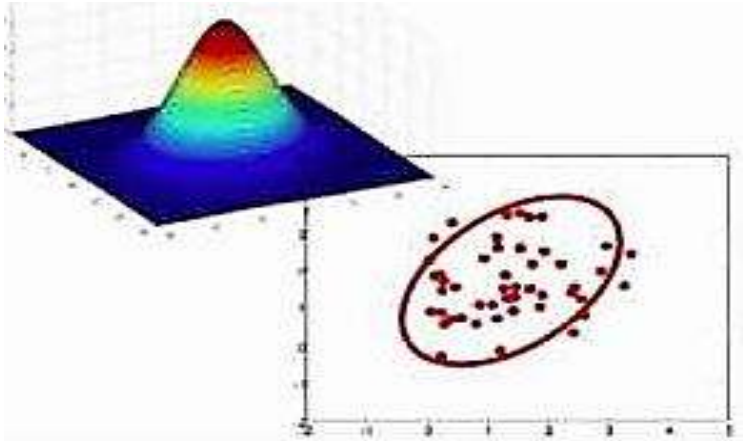
# Clasificación Estadística Paramétrica

Supongamos que las clases responden a distribuciones normales, o sea que las densidades de probabilidad condicionales de cada clase tienen la forma

$$p(\mathrm{x}|\omega_i) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_i|^{\frac{1}{2}}} \exp\Big(-\frac{1}{2}(\mathrm{x}-\boldsymbol{\mu}_i)\Sigma_i^{-1}(\mathrm{x}-\boldsymbol{\mu}_i)^T\Big) \quad \forall\, i = 1, \cdots, m\,.$$

siendo

$$\begin{cases} \boldsymbol{\mu}_i = E\left[\mathrm{x}|\omega_i\right] \\ \Sigma_i = \mathrm{Cov}[\mathrm{x}|\omega_i] = E\left[(\mathrm{x}-\boldsymbol{\mu}_i)(\mathrm{x}-\boldsymbol{\mu}_i)^T|\omega_i\right] \end{cases}$$

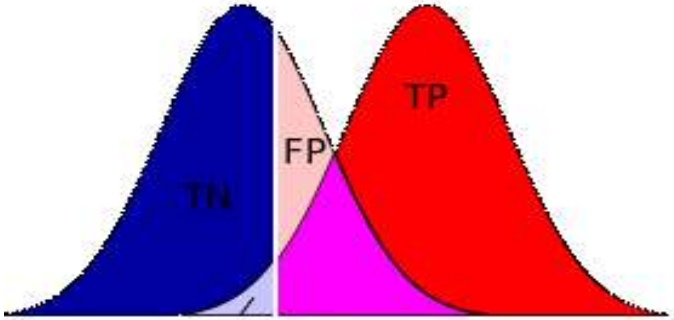


# Clasificación Estadística Paramétrica: Regla del mínimo error

la regla de decisión correspondiente resulta:

$$\text{Asignar } \mathrm{x} \to \omega_j \quad \Longleftrightarrow \quad P(\omega_j|\mathrm{x}) = \max_{1 \le k \le m} P(\omega_k|\mathrm{x})$$

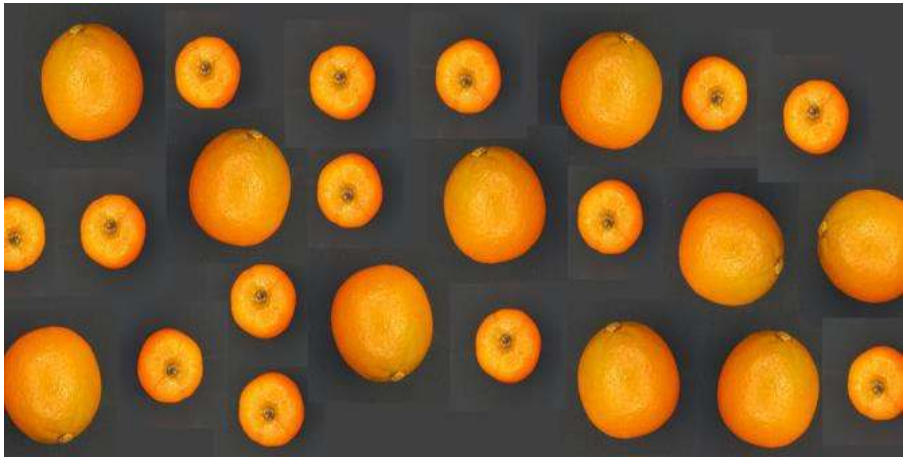
Observar que si asumimos que se asigna $\mathrm{x} \to \omega_j$, la probabilidad condicional de error $\epsilon(\mathrm{x})$ viene dada por




$$\epsilon(\mathrm{x}) = 1 - P(\omega_j|\mathrm{x})\,.$$

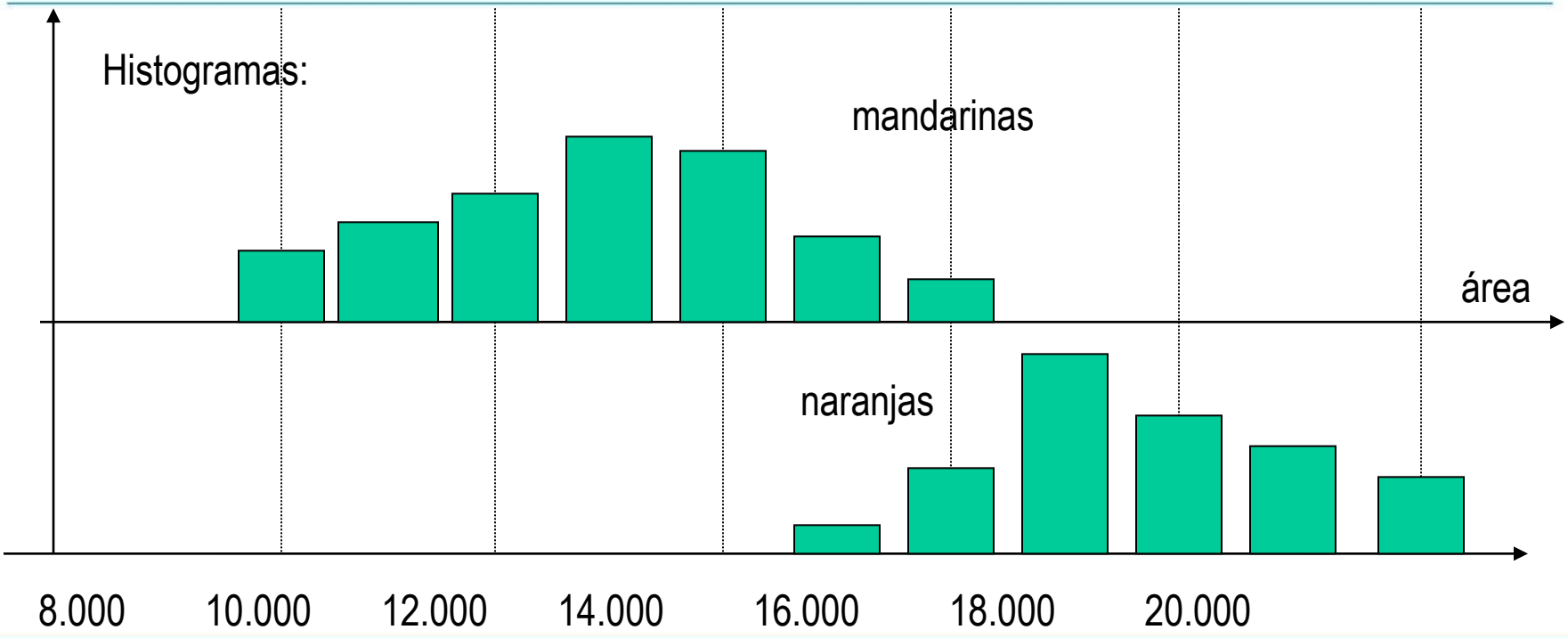
# Clasificación Estadística Paramétrica: un ejemplo

Determinar si es naranja o mandarina de acuerdo con el tamaño (área)



| | | | |
|---|---|---|---|
| Naranja-01 | 19.327 | Mandarina-01 | 13.221 |
| Naranja-02 | 18.265 | Mandarina-02 | 14.987 |
| Naranja-03 | 17.456 | Mandarina-03 | 15.321 |
| Naranja-04 | 19.341 | Mandarina-04 | 15.987 |
| Naranja-05 | 16.342 | Mandarina-05 | 16.345 |
| Naranja-06 | 16.987 | Mandarina-06 | 15.965 |
| Naranja-07 | 17.001 | Mandarina-07 | 16.341 |
| : | 19.056 | : | |
| Naranja-75 | 15.900 | Mandarina-50 | 13.439 |

# Clasificación Estadística Paramétrica: un ejemplo

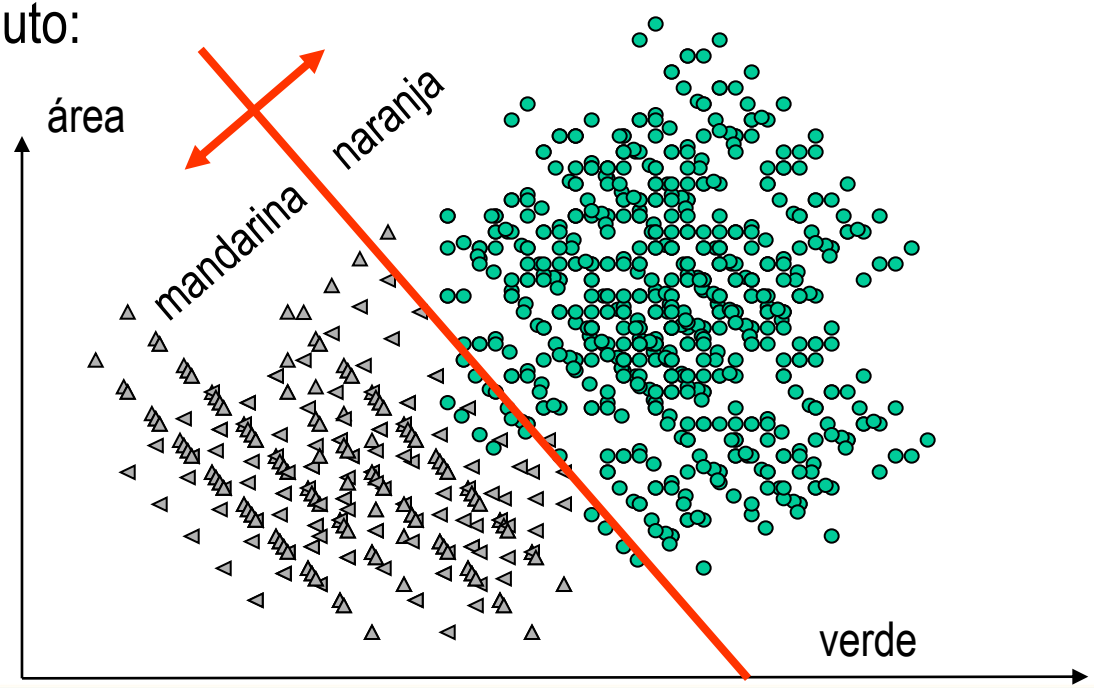


# Clasificación Estadística Paramétrica Multivariada

Podemos tener más de un atributo:

Verde = 23.6%

Verde = 46%



# Clasificación Estadística Paramétrica Multivariada

Supongamos que las probabilidades a priori y las covarianzas son constantes en las clases

$$\Sigma_i = \Sigma_j = \Sigma \quad \text{y} \quad P(\omega_i) = P(\omega_j) \quad \forall 1 \leq i \neq j \leq m$$

Definimos la norma y distancias asociadas a $\Sigma$ como antes:

$$\|\mathbf{x}\|^2 = \mathbf{x}\Sigma^{-1}\mathbf{x}^T \quad \Rightarrow \quad d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\| \quad \forall\, \mathbf{x}, \mathbf{x}' \in \Omega$$

Esta distancia se conoce como *distancia de Mahalanobis* asociada a la covarianza $\Sigma$. Notar que en el caso particular que $\Sigma = \mathbf{I}$ está distancia coincide con la distancia cuadrática Euclidiana.

Por lo tanto la regla del mínimo error se reduce a asignar el patrón $\mathbf{x}$ a la clase cuyo vector medio sea el más cercano según la distancia de Mahalanobis (*Regla de decisión por media más cercana*):

$$\text{Asignar } \mathbf{x} \rightarrow \omega_j \quad \Longleftrightarrow \quad d(\mathbf{x}, \boldsymbol{\mu}_j) = \min_{1 \leq k \leq m} d(\mathbf{x} - \boldsymbol{\mu}_k)$$

# Clasificación Estadística Paramétrica Multivariada

Ahora analizaremos el problema general de decisión entre dos clases

$$\Rightarrow \quad m = 2, \quad \Sigma_1 \neq \Sigma_2$$

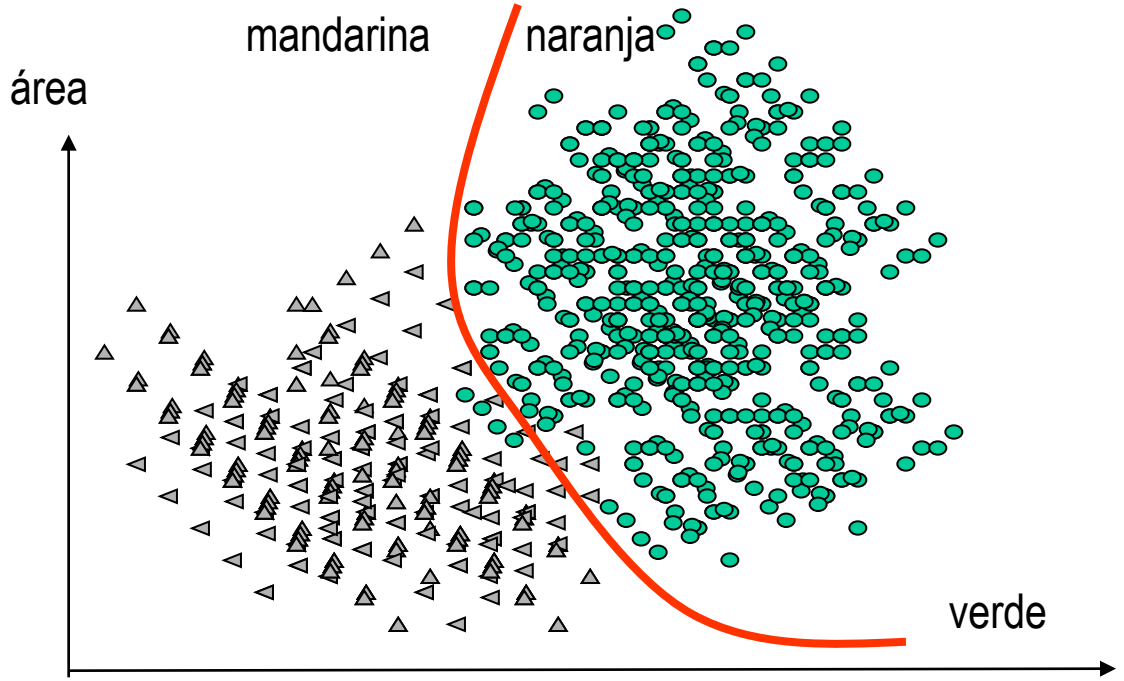
la ecuación

$$f(\mathrm{x}) = \|\mathrm{x} - \boldsymbol{\mu}_1\|_1 - \|\mathrm{x} - \boldsymbol{\mu}_2\|_2 + C_2 - C_1 = 0$$

define la superficie de separación, conocida como *superficie discriminante*, entre las regiones asociadas a cada una de las clases $\omega_1$ y $\omega_2$. En general esta superficie es cuadrática ya que su ecuación resulta:

$$\underbrace{\mathrm{x}^T\left(\Sigma_1^{-1} - \Sigma_2^{-1}\right)\mathrm{x}}_{\text{cuadrático}} - \underbrace{2\mathrm{x}^T\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 - \Sigma_2^{-1}\boldsymbol{\mu}_2\right)}_{\text{lineal}} + \underbrace{\left(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2 + C_2 - C_1\right)}_{\text{constante}} = 0\,.$$

# Clasificación Estadística Paramétrica Multivariada



# Clasificación Bayesiana

Definamos un costo de decisión $\rho_{ij}$ $1 \leq i, j \leq n$ asociado con la decisión de asignar a la clase $\omega_j$ un patrón x que pertenece a la clase $\omega_i$.

Notar que

$\rho_{ii}$ costo de una decisión correcta, en general se define como 0

$\rho_{ij}$ es en general diferente de $\rho_{ji}$

$\rho_{ij} \geq 0$

# Clasificación Bayesiana

El costo medio de clasificar un patrón $\mathbf{x} \in \Gamma_j$ como perteneciente a la clase $\omega_j$ es

$$r_j(\mathbf{x}) = \sum_{i=1}^{m} \rho_{ij} P(\omega_i|\mathbf{x})$$

Por lo tanto el costo para toda la región $\Gamma_j$ se obtiene integrando sobre todos los valores posibles con sus correspondientes probabilidades de observación:

$$R_j = \int_{\Gamma_j} r_j(\mathbf{x}) p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

Finalmente el costo total de nuestro sistema de decisión viene dado por

$$R = \sum_{j=1}^{m} R_j = \sum_{j=1}^{m} \int_{\Gamma_j} \left( \sum_{i=1}^{m} \rho_{ij} P(\omega_i|\mathbf{x}) \right) p(\mathbf{x}) \mathrm{d}\mathbf{x}$$

# Clasificación Bayesiana

De modo que podemos concluir que el costo total será minimizado si el espacio de observación se particiona de manera tal que si $\mathbf{x} \in \Gamma_j$ se tenga

$$\sum_{i=1}^{m} \rho_{ij} P(\omega_i|\mathbf{x}) \leq \sum_{i=1}^{m} \rho_{ik} P(\omega_i|\mathbf{x}) \quad \forall\, k \neq j$$

Hemos llegado por lo tanto a la llamada *Regla de Decisión de Mínimo Costo de Bayes*, que establece

$$\text{Asignar } \mathbf{x} \rightarrow \omega_j \quad \Longleftrightarrow \quad \sum_{i=1}^{m} \rho_{ij} p(\mathbf{x}|\omega_i) P(\omega_i) = \min_{1 \leq k \leq m} \sum_{i=1}^{m} \rho_{ik} p(\mathbf{x}|\omega_i) P(\omega_i)$$

donde en en la última ecuación hemos usado la identidad de Bayes: $P(\omega_i|\mathbf{x})p(\mathbf{x}) = p(\mathbf{x}|\omega_i)P(\omega_i)$.

# Clasificación Bayesiana

Observar que el teorema de Bayes permite estimar la probabilidad a-posteriori a partir de la probabilidad a-priori, de la probabilidad condicional (pdf o likelihood), y de *la* **distribución marginal de los patrones** *p(*x*)*.

Por lo tanto, es básicamente adecuado cuando el espacio de atributos puede representarse en términos de frecuencias relativas. Cuando en el espacio de atributos tenemos variables cuantitativas, es necesario «nominalizarlas» para poder transformar los valores cuantitativos en frecuencias relativas («probabilidades»).

# Regresión Logística

Pese a su nombre, es un método de clasificación. La regresión es respecto a la probabilidad de que un patrón pertenezca a una determinada clase.

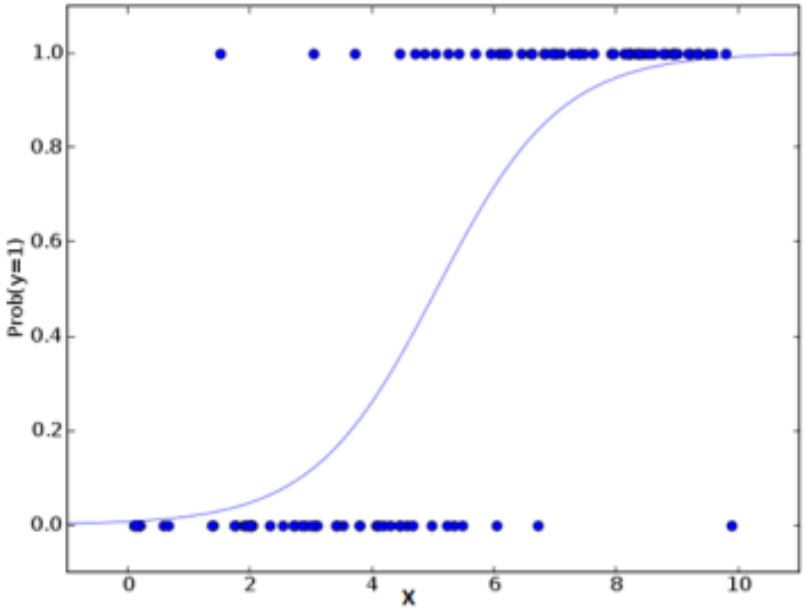
Al igual que con el caso paramétrico, se elige la clase más probable dada la observación, pero la probabilidad no se calcula en base a parámetros estadísticos sino con una función de regresión.

La regresión lineal es inadecuada para probabilidades, dado que éstas no tienen sentido o significado para valores fuera del intervalo [0, 1]. Por ello se usa la denominada función logística.

# Regresión Logística

La función logística se basa en las «chances» (odds) de x (el cociente entre la probabilidad de que x ocurra sobre la probabilidad de que no).

A diferencia del clasificador Bayesiano, en la regresión logística el vector de atributos es cuantitativo (continuo). Si tenemos variables nominales es necesario cuantificarlas.



# Árboles de Decisión

La inducción de árboles de decisión es una de las aplicaciones exitosas del ML. Un árbol de decisión toma una determinada situación como entrada, y produce una “decisión” como salida, luego de realizar un grupo de tests.

Cada nodo no hoja del árbol tiene asociado un atributo, mientras que cada hoja tiene asociado un resultado. Cada arco tiene asociado un posible valor del atributo del nodo del cual parte.

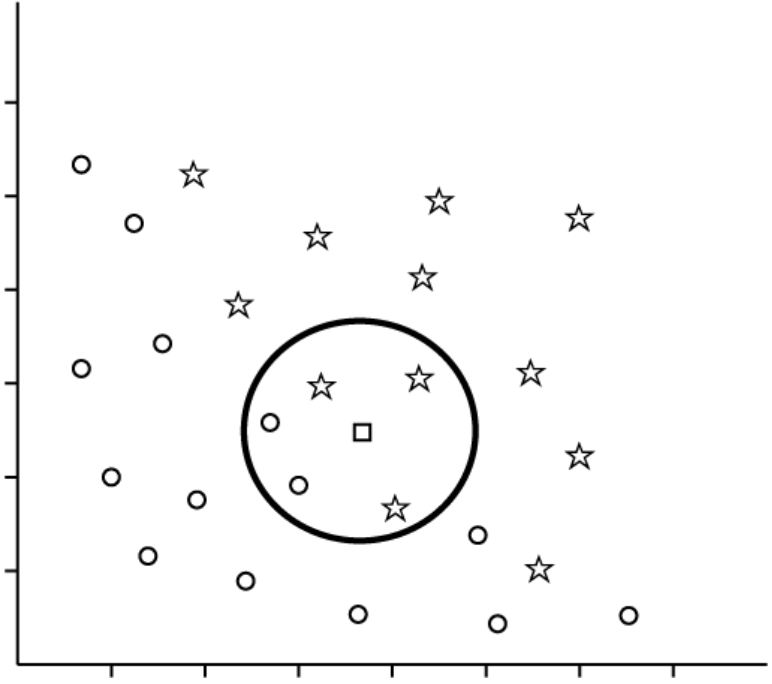
A partir de un conjunto de ejemplos (valores de los atributos, y decisión correcta esperada) se puede “inducir” un árbol de decisión.

# Clasificación: K vecinos más cercanos

Determina el conjunto de K casos conocidos más cercanos y decide en función de la categoría de la mayoría.

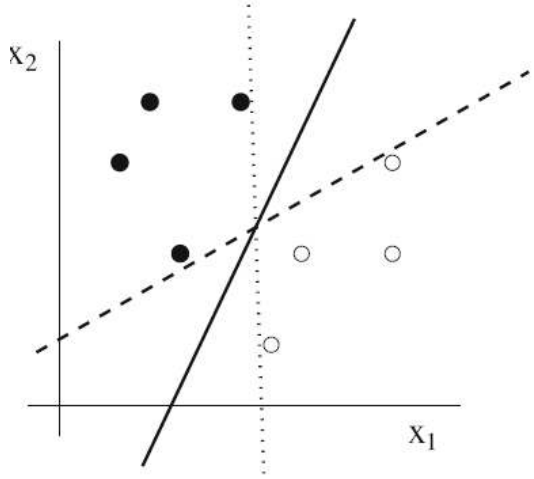
Es conveniente trabajar con valores estandarizados. Los valores nominales deben cuantificarse.

El valor de K determina el balance entre sesgo y variancia.



# Clasificación: Support Vector Machines

Se basa en dos ideas fundamentales:




Puntos etiquetados que no sean linealmente separables se pueden mapear a un espacio de más dimensiones (usando una función kernel) donde tal separación es posible. El kernel a su vez puede facilitar la representación.

Dado un conjunto de puntos etiquetados linealmente separables, es posible encontrar un hiperplano de separación óptima (que maximice el margen de separación) y por lo tanto que sea más inmune a ruido y overfitting.

# Clasificación: Redes Neuronales

El punto de partida histórico fueron las funciones discriminantes lineales, las cuales dieron origen al primer modelo de aprendizaje, conocido como **perceptrón**.

Luego fueron agregándose diferentes niveles de complejidad, hasta llegar finalmente al actual deep learning.

Cada «capa» de la red neuronal realiza tareas sencillas de clasificación, de nivel de abstracción creciente.

# Clasificación: Redes Neuronales

Las fronteras entre regiones son (hiper)superficies, llamadas *superficies de decisión.*

En el caso más sencillo (pero suficientemente poderoso), las hipersuperficies consisten en hiperplanos, cuya ecuación genérica en el espacio de los patrones es

$$w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + w_{n+1} = 0$$

De esa manera, cada clase tendrá una una *función discriminante lineal*

$$d_k(\mathbf{x}) = w_{1,k} x_1 + w_{2,k} x_2 + \cdots + w_{n,k} x_n + w_{n+1,k}$$

La expresión

$$d_k(\mathbf{x}) = w_{1,k}x_1 + w_{2,k}x_2 + \cdots + w_{n,k}x_n + w_{n+1,k}$$

puede pensarse como

$$\mathbf{w}_k^T\mathbf{x}$$

donde $\mathbf{w}_k^T = [w_{1,k}\ w_{2,k}\ \cdots\ w_{n,k}\ w_{n+1,k}]$ es el *vector de pesos (aumentado)*,

y $\mathbf{x}^T = [x_1\ x_2\ \cdots\ x_n\ 1]$ es el *patrón aumentado.*

# Clasificación: Redes Neuronales

Un patron $\mathbf{x}$ pertenece a la clase $\omega_j$ cuando la distancia entre $\mathbf{x}$ y $\mathbf{z}_j$ es menor que al centro de cualquier otra clase.

Una primera aproximación para este clasificador consiste en utilizar distancia Euclídea al cuadrado:

$$d(\mathbf{x}, \mathbf{z}_k)^2 = |\mathbf{x} - \mathbf{z}_k|^2 = (\mathbf{x} - \mathbf{z}_k)^T(\mathbf{x} - \mathbf{z}_k) = \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{z}_k + \mathbf{z}_k^T\mathbf{z}_k$$

El término $\mathbf{x}^T\mathbf{x}$ es independiente de la clase $k$ y por lo tanto puede ser eliminado.

Queremos ahora encontrar el $\mathbf{z}_k$ tal que $-2\mathbf{x}^T\mathbf{z}_k + \mathbf{z}_k^T\mathbf{z}_k$ sea mínimo.

# Clasificación: Redes Neuronales

Minimizar $-2\mathbf{x}^T\mathbf{z}_k + \mathbf{z}_k^T\mathbf{z}_k$ es lo mismo que maximizar $\mathbf{x}^T\mathbf{z}_k - \frac{1}{2}\mathbf{z}_k^T\mathbf{z}_k$

Esto puede expresarse como una función discriminante

$$d_k(\mathbf{x}) = \mathbf{x}^T\mathbf{z}_k - \frac{1}{2}\mathbf{z}_k^T\mathbf{z}_k = \mathbf{x}^T\mathbf{z}_k - \frac{1}{2}|\mathbf{z}_k|^2 = \mathbf{x}^T\mathbf{w}_k$$

donde el vector aumentado de pesos $\mathbf{w}_k$ es $\mathbf{w}_k = \left[ z_{k,1} \;\; z_{k,2} \;\; \cdots \;\; z_{k,n} \;\; \frac{1}{2}|\mathbf{z}_k|^2 \right]$

El hiperplano de decisión entre las clases $j$ y $k$, entonces, queda definida como

$$d_{jk}(\mathbf{x}) = \mathbf{x}^T(\mathbf{z}_j - \mathbf{z}_k) - \frac{1}{2}(|\mathbf{z}_j|^2 - |\mathbf{z}_k|^2) = 0$$

Es fácil ver que este hiperplano es perpendicular al vector que une $\mathbf{z}_j$ con $\mathbf{z}_k$, y que cruza dicho vector exactamente en el punto medio entre ambos centros de clase.

# Clasificación: Redes Neuronales

Dado un discriminante lineal $d_k(\mathbf{x}) = w_{1,k}x_1 + w_{2,k}x_2 + \cdots + w_{n,k}x_n + w_{n+1,k} = \mathbf{w}^T\mathbf{x}$, el cual puede pensarse como el producto escalar entre un patrón aumentado y un vector de pesos aumentado.

El problema del entrenamiento consiste en encontrar un vector de pesos adecuado.

Es más adecuado representar este problema en el *espacio de los pesos*, donde cada valor de $\mathbf{w}$ es representado con un punto, y viceversa.

El espacio de los pesos es un espacio $(n+1)$-dimensional en el cual las coordenadas son $w_1, w_2, \cdots, w_{n+1}$.

La región en este espacio donde $\mathbf{w}^T\mathbf{x} = 0$ es también un hiperplano, que siempre pasa por el origen (sin traslación).

# Clasificación: Redes Neuronales

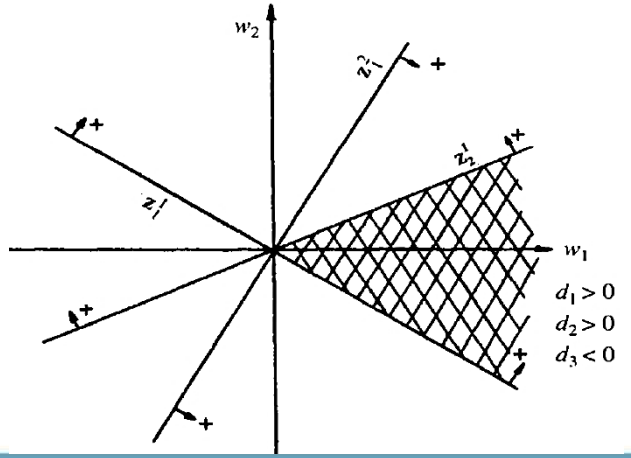
Dado un prototipo $\mathbf{z}_k^i$ de la clase $k$, todo hiperplano $\mathbf{w}$ tal que $\mathbf{w}^T\mathbf{z}_k^i > 0$ probablemente clasificará a $\mathbf{z}_k^i$ como perteneciente a $\omega_k$.

Supongamos tener dos clases $\omega_1, \omega_2$ linealmente separables, cada una con $N_1, N_2$ prototipos.

En ese caso queremos encontrar un vector de pesos $\mathbf{w}$ tal que

$$\mathbf{w}^T\mathbf{z}_1^i > 0,\ \forall \mathbf{z}_1^i \in \omega_1, i = 1, \cdots, n_1$$

$$\mathbf{w}^T\mathbf{z}_2^i < 0,\ \forall \mathbf{z}_2^i \in \omega_2, i = 1, \cdots, n_2$$



# Clasificación: Redes Neuronales

Cuando ocurre un error en el entrenamiento, ello se debe a que dado el vector de pesos $\mathbf{w}_k$, para alguna clase, con dicho vector algún prototipo queda mal clasificado.

Corregir el error implica colocar a dicho prototipo en la clasificación adecuada, es decir, encontrar un nuevo $\mathbf{w}'_k$ en la zona adecuada del espacio de los pesos.

El procedimiento habitual es mover $\mathbf{w}_k$ en dirección perpendicular al $\mathbf{w}_k$ mal clasificado (hacia la zona positiva si fue un falso negativo, o hacia la zona negativa si fue un falso positivo).

Eso se logra simplemente sumando (restando) el prototipo mal clasificado al vector de pesos. Se utiliza un *factor de corrección* $c$ para regular la convergencia del procedimiento.

$$\mathbf{w}'_k = \mathbf{w}_k + (-)c\mathbf{z}_k$$

# Clasificación: Redes Neuronales

Si más de un prototipo fue mal clasificado, se elige uno entre todos al azar.

El procedimiento se *itera* hasta que todos los prototipos queden correctamente clasificados.

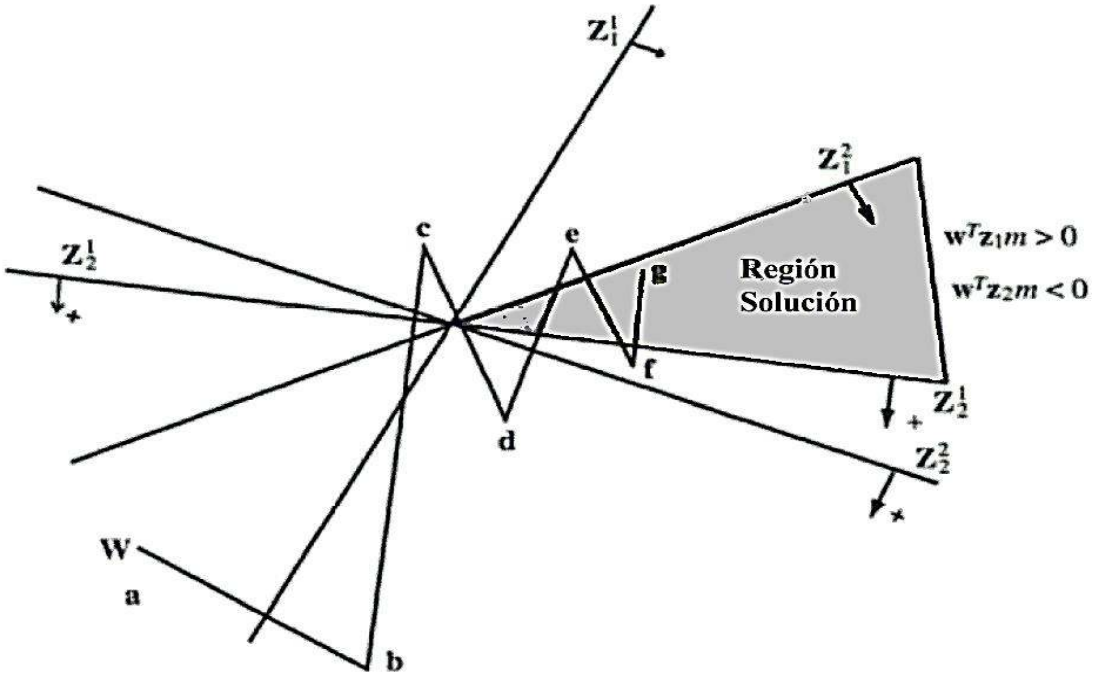
Es importante observar que existe un teorema que demuestra la convergencia del procedimiento anterior, siempre que $c$ sea positivo, y las clases sean separables.

Modificar el valor de $c$ equivale a multiplicar los prototipos sin alterar su separabilidad.

Los diferentes criterios para la elección de $c$ (fijo, fraccional, absoluto, etc.) determinan la velocidad de convergencia.

# Clasificación: Redes Neuronales

Un ejemplo de aprendizaje por medio de la regla de la corrección del error. El valor de c se va decrementando linealmente.



# Por qué necesitamos otras redes neuronales?

Redes profundas: mayor cantidad de niveles de abstracción.

Redes convolucionales: capacidad de encontrar abstracciones más poderosas.

Redes adversarias: utilizar criterios evolutivos para encontrar soluciones

...