IE3038 – DISEÑO E INNOVACIÓN EN INGENIERÍA 1

INTRODUCCIÓN A *MACHINE LEARNING*

Luis Alberto Rivera

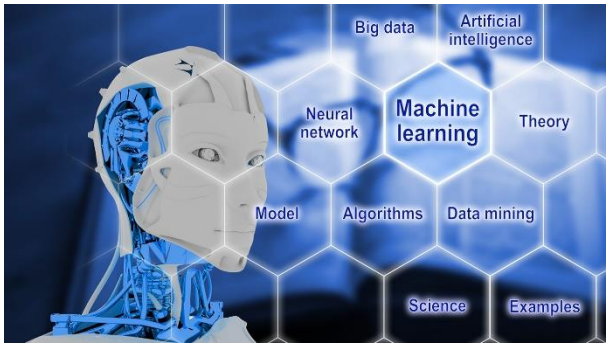
1er ciclo, 2023

Aprendizaje Automático/de Máquina (*Machine Learning*) y Reconocimiento de Patrones (*Pattern Recognition*)

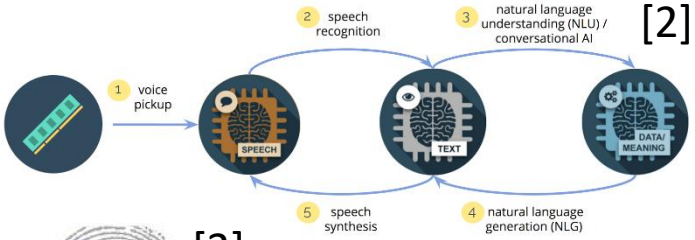
- Nace como subdisciplina de la Inteligencia Artificial.
- ¿Pueden las computadoras aprender únicamente basándose en datos, sin ser programadas explícitamente?
- ¿Personas y máquinas trabajando de la mano?

Aplicaciones de *Machine Learning* y *Pattern Recognition*

- Análisis y reconocimiento de imágenes, sonido, texto.
- Diagnósticos médicos.
- Clasificación y predicción de eventos.
- Minería de datos *(Data Mining).*
- Robótica, Finanzas, Telecomunicaciones, Procesos Industriales, etc.

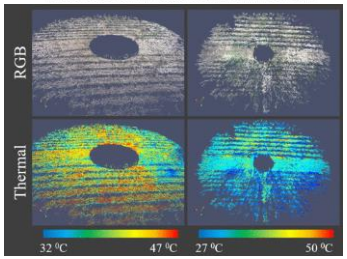


[1]

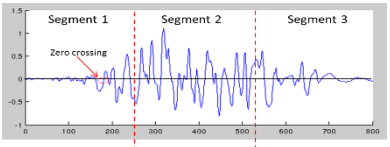


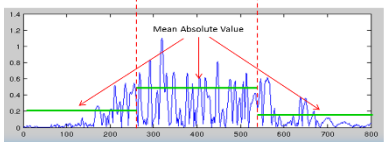
[2]


[3]



[4]





[5]

Patrones, Características/Rasgos (*Features*)

Ejemplos

Área	Características (*features*)
Análisis de señales (sonido, bioeléctricas, etc.)	Pendientes, amplitud máxima, potencia, # de cambios de signo, componentes de frecuencia, etc.
Reconocimiento de escritura a mano	Formas, trazos, puntos clave (esquinas, lazos), etc.
Identificación de rostros	Formas, distancia entre ojos, lunares, etc.

Característica: propiedad medible de un objeto / individuo / fenómeno observado; comúnmente numérica.

Aprendizaje Supervisado vs. No Supervisado

En la práctica, lo que típicamente se tiene son muestras (conjuntos de datos, comúnmente multidimensionales), y algún conocimiento general / vago de la situación.

Esas muestras se pueden usar para diseñar o entrenar un clasificador, para hacer regresiones, para hacer minería de datos, etc.

Si conocemos las etiquetas de las muestras, es decir, el "estado de la naturaleza" a la que pertenecen las muestras, tenemos un problema de **Aprendizaje Supervisado**.

De lo contrario, si no conocemos las etiquetas, tratamos con un problema de **Aprendizaje No Supervisado**.

Aprendizaje Supervisado

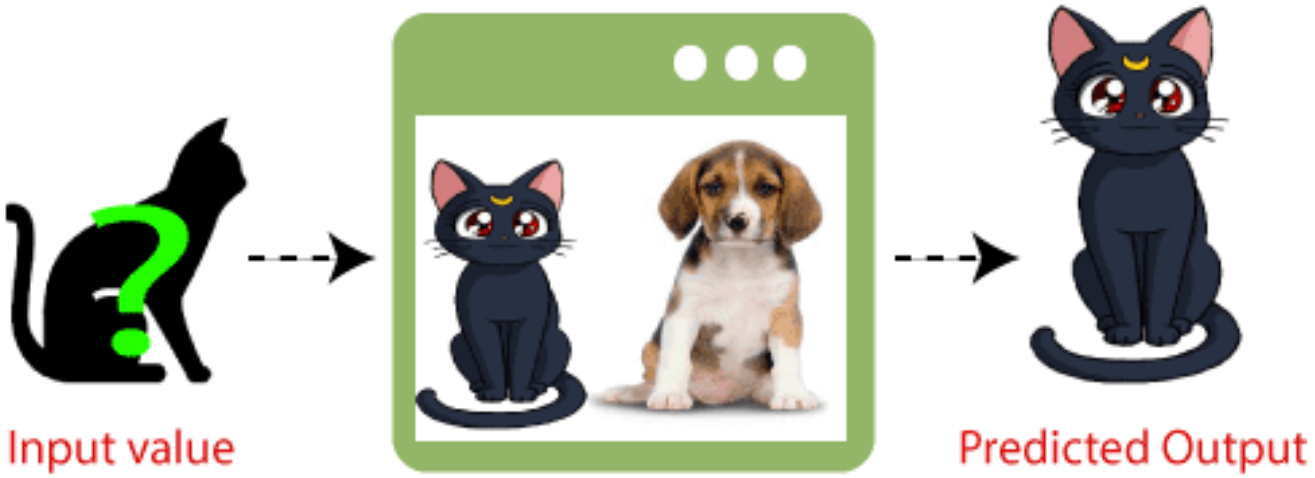
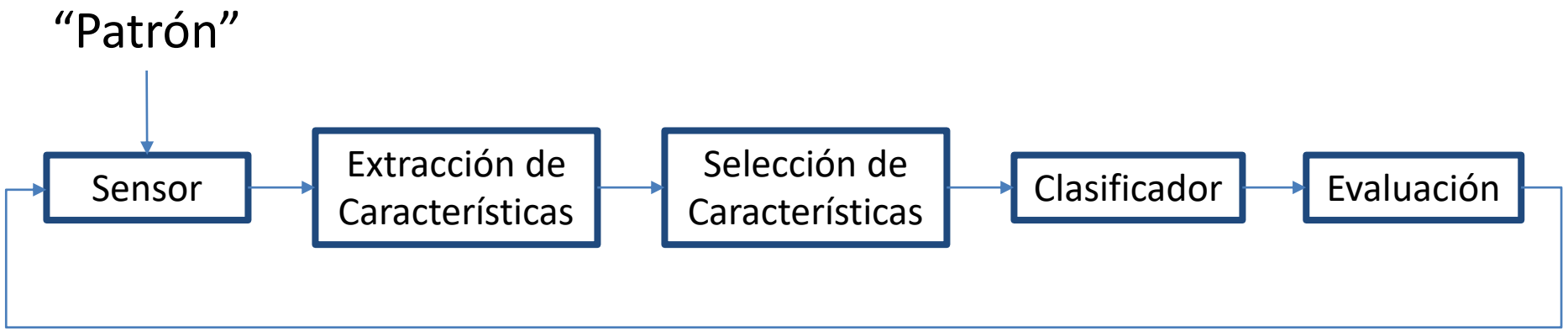


Figura tomada de: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Clasificación



¿Extracción de características?

¿Cuáles son las "mejores"?

¿Separación de las clases/categorías? Lineal vs. no lineal

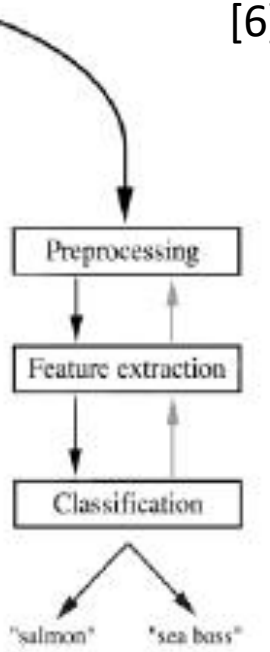
Evaluación de desempeño: criterios, métricas

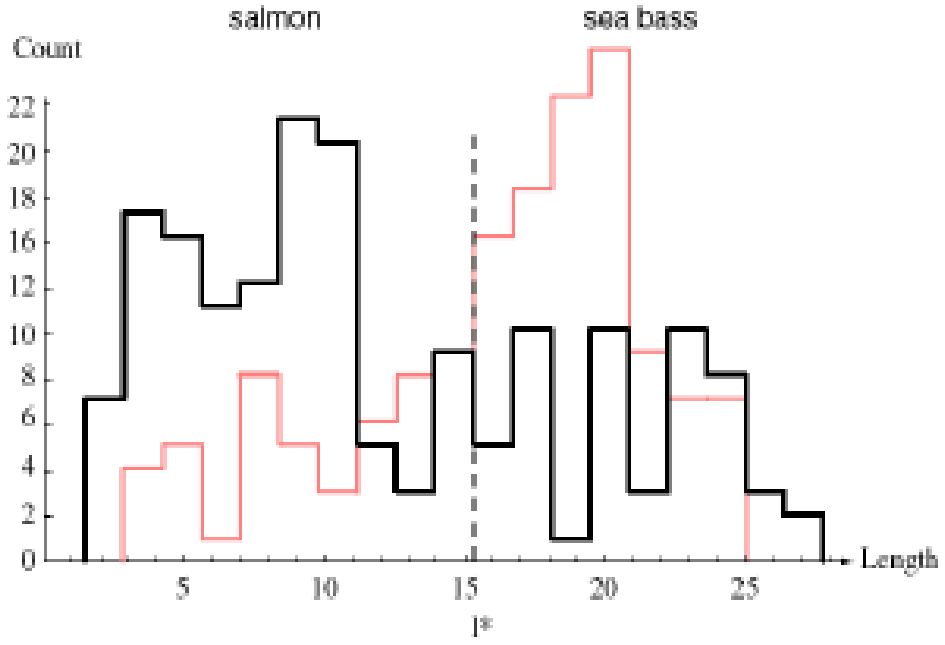
Ejemplo

Clasificador: Salmón vs. Lubina

- Usando longitud

[6]





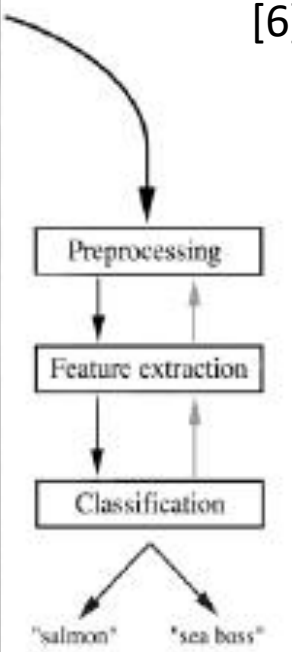
¿Cómo escogemos el umbral de decisión? ¿Riesgo?
¿Cuál es el error esperado en la clasificación?

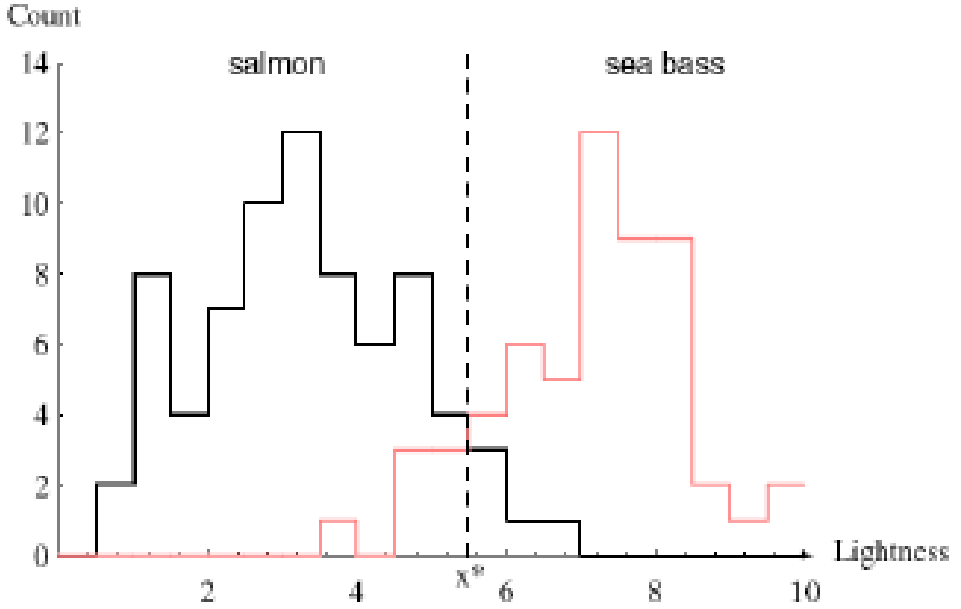
Ejemplo

Clasificador: Salmón vs. Lubina

- Usando claridad

[6]





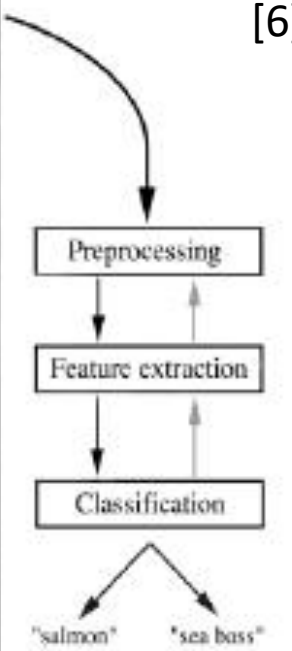
¿Cuál es el error esperado en la clasificación ahora?

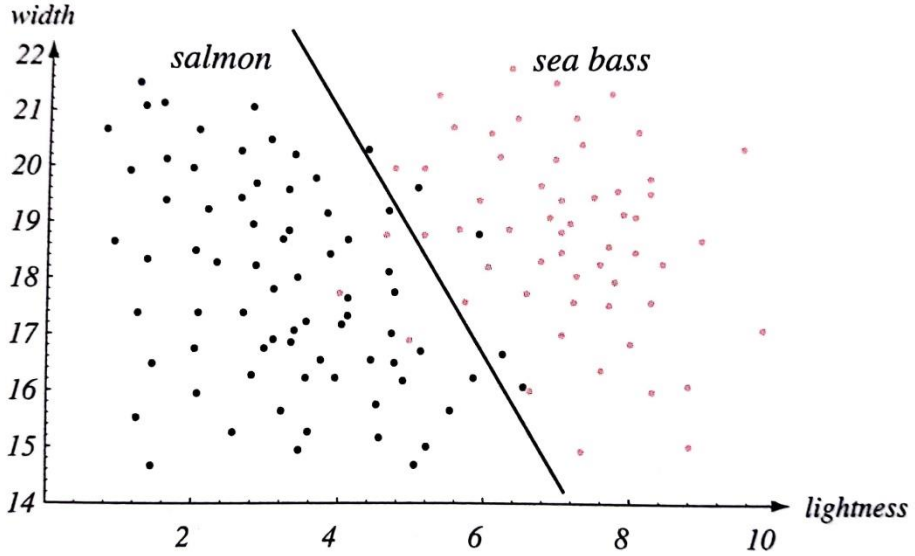
Ejemplo

Clasificador: Salmón vs. Lubina

- Usando 2 características (vector 2-dimensional)

[6]





Ya no tenemos un umbral unidimensional, sino una recta fronteriza en el plano.

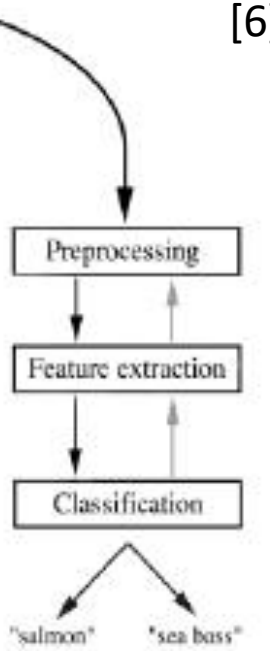
¿Error esperado?

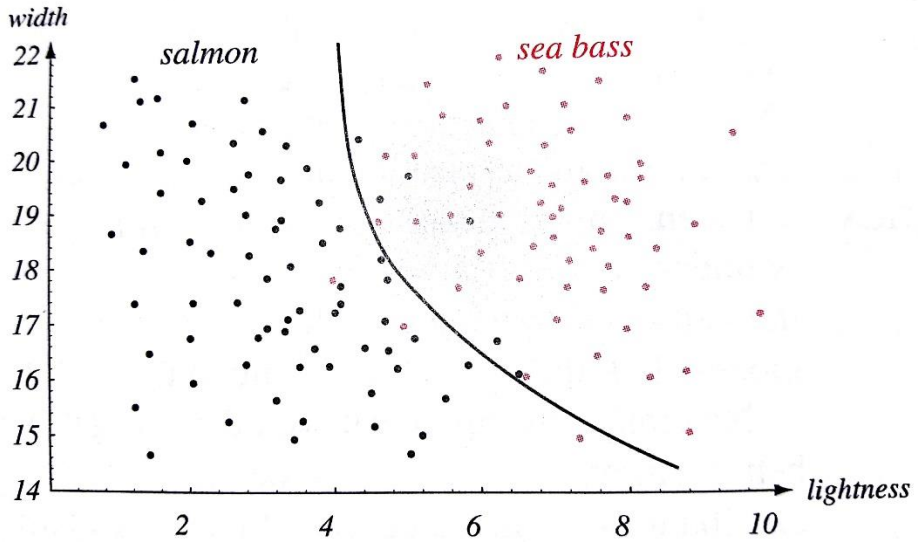
Ejemplo

Clasificador: Salmón vs. Lubina

- Usando 2 características (vector 2-dimensional)

[6]





La frontera no necesariamente tiene que ser una recta...

¿Error esperado?

Entrenar, Validar, Evaluar

En situaciones prácticas, típicamente contamos con datos (muestras) recolectados a través de sensores u otros dispositivos.

Podemos dividir los datos (vectores de características) en conjuntos de entrenamiento, validación y evaluación.

El primer conjunto de datos nos sirve para entrenar nuestro clasificador, luego se puede validar el mismo, para finalmente poder clasificar muestras nuevas (evaluación).

Entrenar, Validar, Evaluar

Validación Cruzada (*cross validation*):

Tomamos un porcentaje de los datos para entrenar el clasificador (por ejemplo, 90%).

Validamos con el resto de los datos.

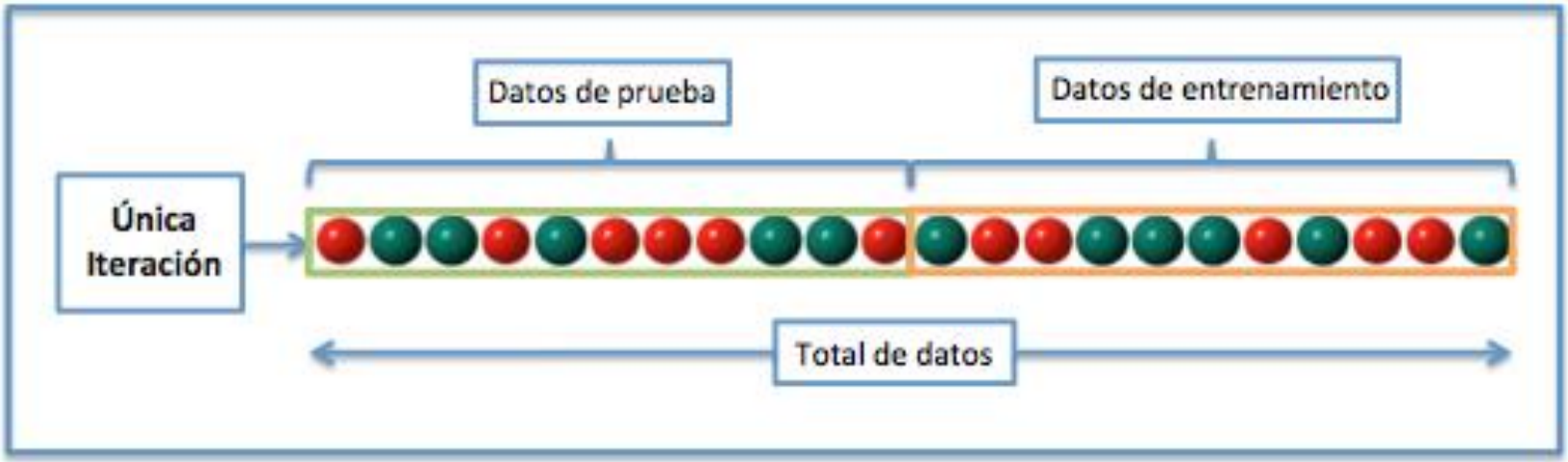


Figura tomada de: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

Entrenar, Validar, Evaluar

Validación Cruzada de k Iteraciones (*k-fold cross validation*):

Repetir el proceso de validación cruzada para k particiones de los datos. Las particiones pueden ser secuenciales o aleatorias.

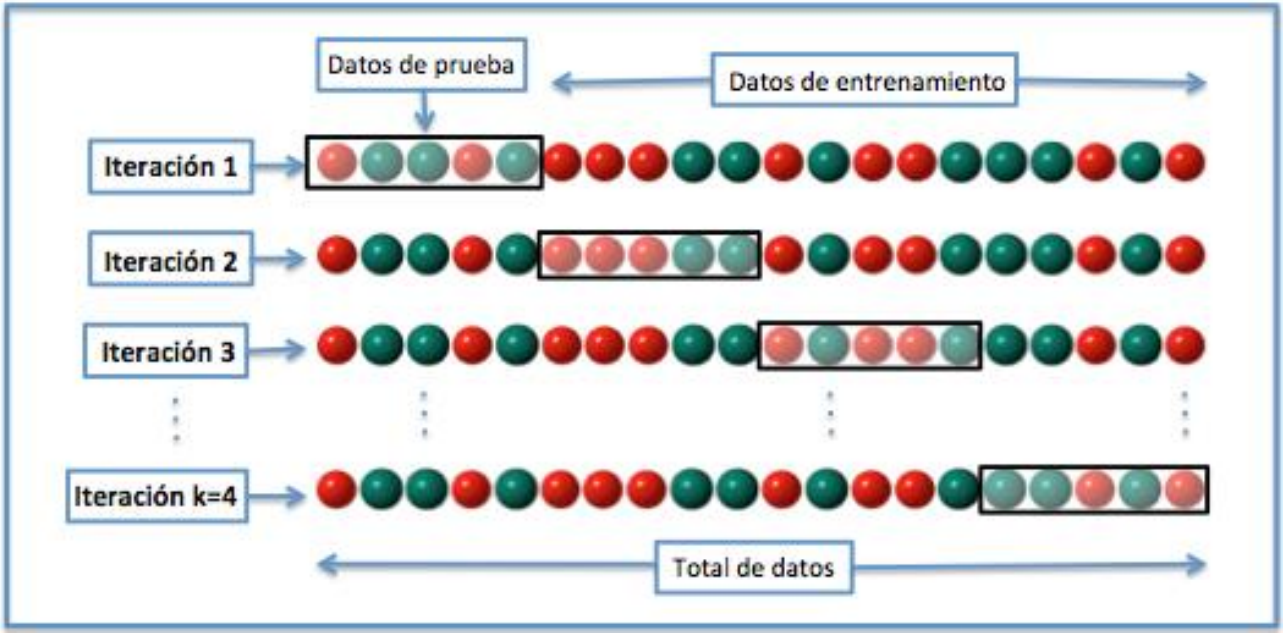


Figura tomada de: https://es.wikipedia.org/wiki/Validaci%C3%B3n_cruzada

Algunos Clasificadores Comunes

- Máquina lineal (funciones discriminantes lineales)
- Máquina de Vectores de Soporte (*Support Vector Machine* – SVM)
- Redes Neuronales Artificiales (*Artificial Neural Networks* – ANN)
- *K-Nearest Neighbor* (KNN)

Funciones Discriminantes Lineales

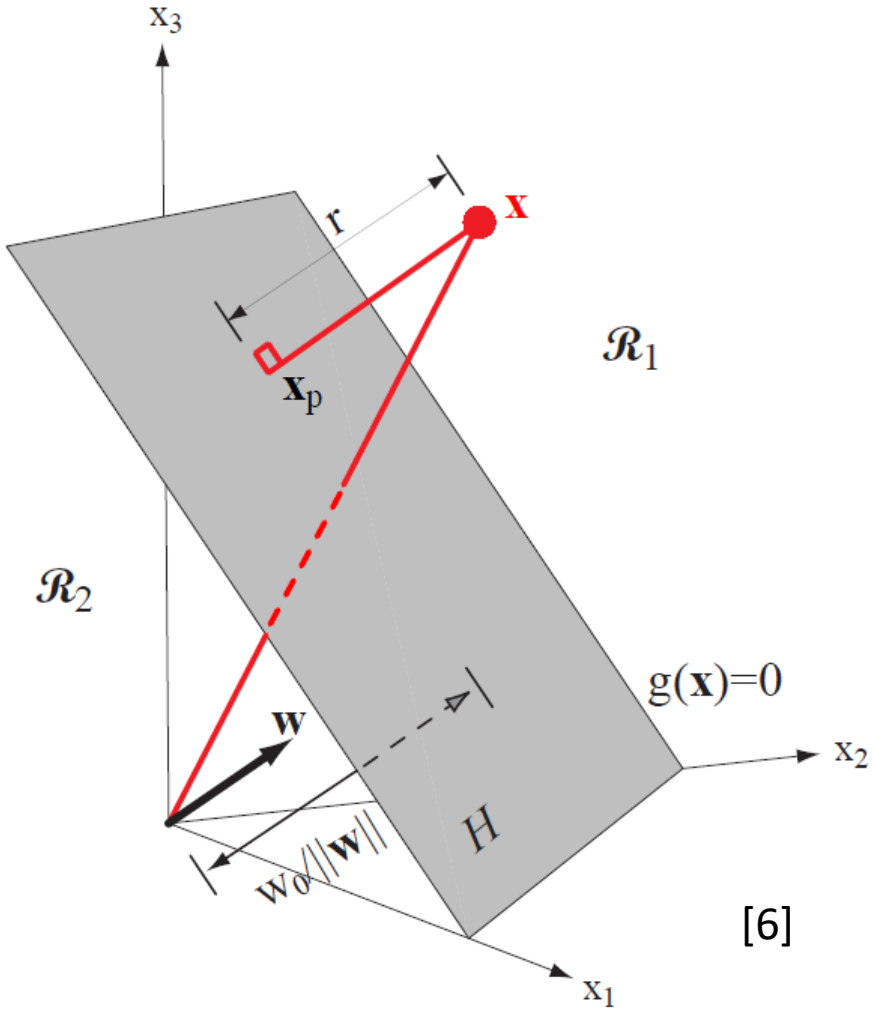
Funciones de la forma:

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0$$

$\boldsymbol{w}$ es un vector de pesos (*weight vector*)

w_0 es un término de sesgo o umbral (*bias, threshold weight*).

Hay diversos algoritmos para encontrar $\boldsymbol{w}$ y w_0.



[6]

Figure 5.2: The linear decision boundary H, where $g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = 0$, separates the feature space into two half-spaces $\mathcal{R}_1$ (where $g(\mathbf{x}) > 0$) and $\mathcal{R}_2$ (where $g(\mathbf{x}) < 0$).

Funciones Discriminantes Lineales

Múltiples categorías:

$$g_i(\boldsymbol{x}) = \mathbf{w}_i^T \boldsymbol{x} + w_{i0} \qquad i = 1, \ldots, c$$

Máquina lineal (*Linear Machine*):

Un vector $\boldsymbol{x}$ se asigna a la clase i si $g_i(\boldsymbol{x}) > g_j(\boldsymbol{x})$ para toda $i \neq j$. En caso de empates, la clasificación se deja indefinida.

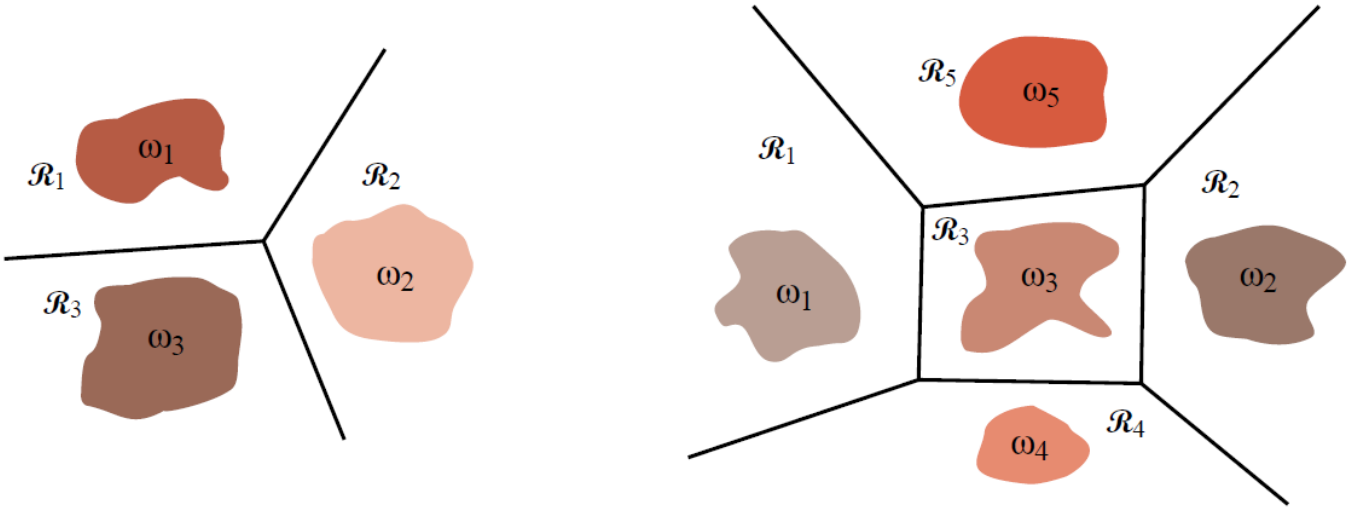


Figure 5.4: Decision boundaries produced by a linear machine for a three-class problem and a five-class problem.

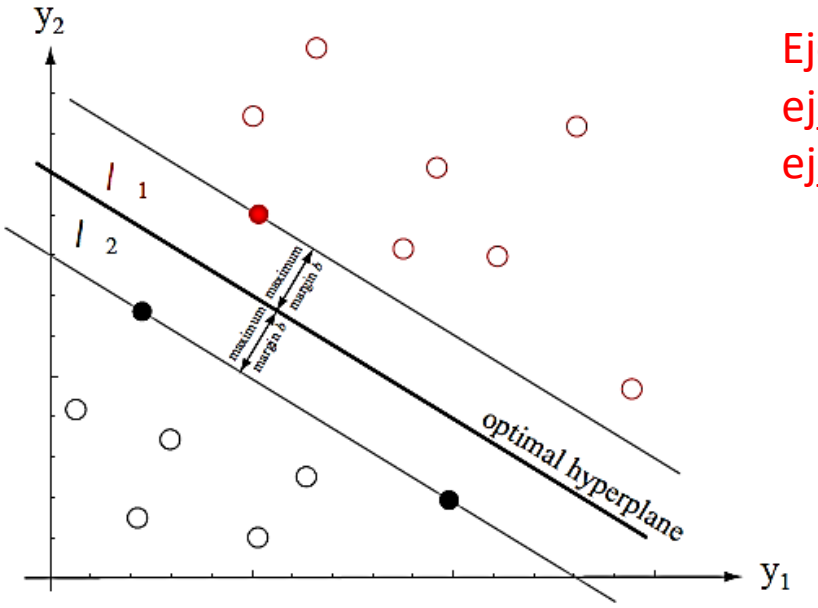
[6]

Máquina de Vectores de Soporte (SVM)

Si las categorías no son linealmente separables en el espacio original ($\boldsymbol{x}$), tal vez lo sean en un espacio de mayor dimensionalidad.

Por medio de un mapeo no lineal $\varphi(\cdot)$ a un espacio de dimensionalidad suficientemente alta, podemos separar las dos categorías por medio de un hiperplano.

El SVM es originalmente un clasificador binario (dos clases).
Existen versiones multi-clase.



[6]

Ejemplos en Matlab:
ej_svm1.m
ej_svm2.m

Figure 5.19: Training a Support Vector Machine consists of finding the optimal hyperplane, i.e., the one with the maximum distance from the nearest training patterns. The support vectors are those (nearest) patterns, a distance b from the hyperplane. The three support vectors are shown in solid dots.

Neuronas Artificiales

- Neurona Artificial (NA) – modelo de una neurona biológica.
- Cada NA recibe señales del "entorno" o de otras NAs. Cuando una NA "se dispara", transmite señales a las NAs conectadas.

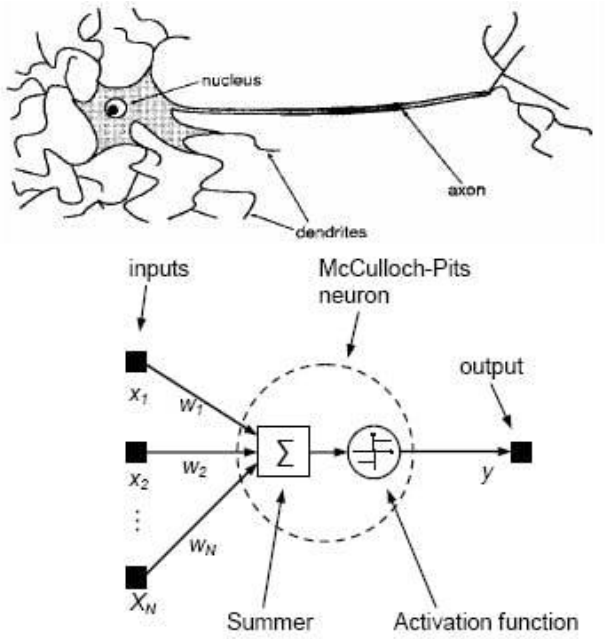


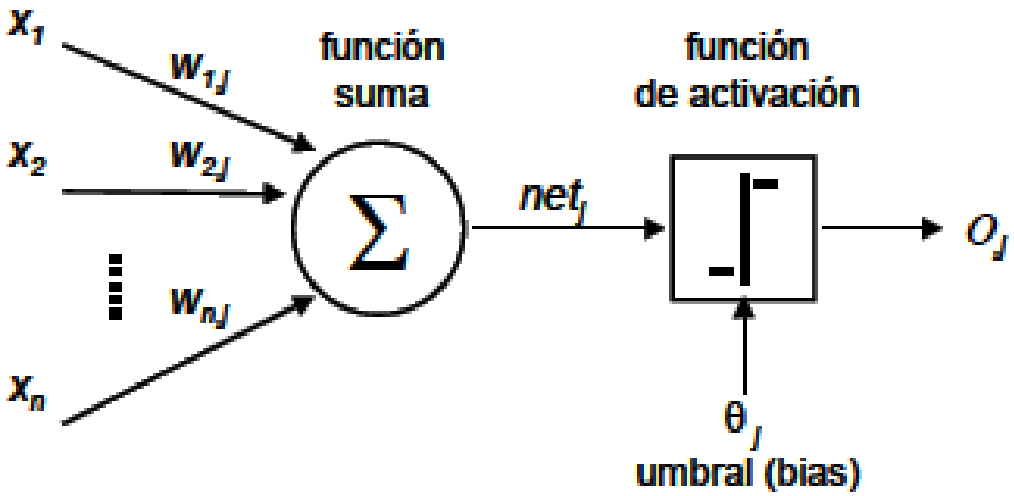
Figura tomada de: https://www.controlglobal.com/control/plcs-pacs/article/11382993/artificial-intelligence-in-process-automation

Neuronas Artificiales

Mapeo no lineal:

$$f_{NA}: \mathbb{R}^n \rightarrow [0,1]$$

o $f_{NA}: \mathbb{R}^n \rightarrow [-1,1]$

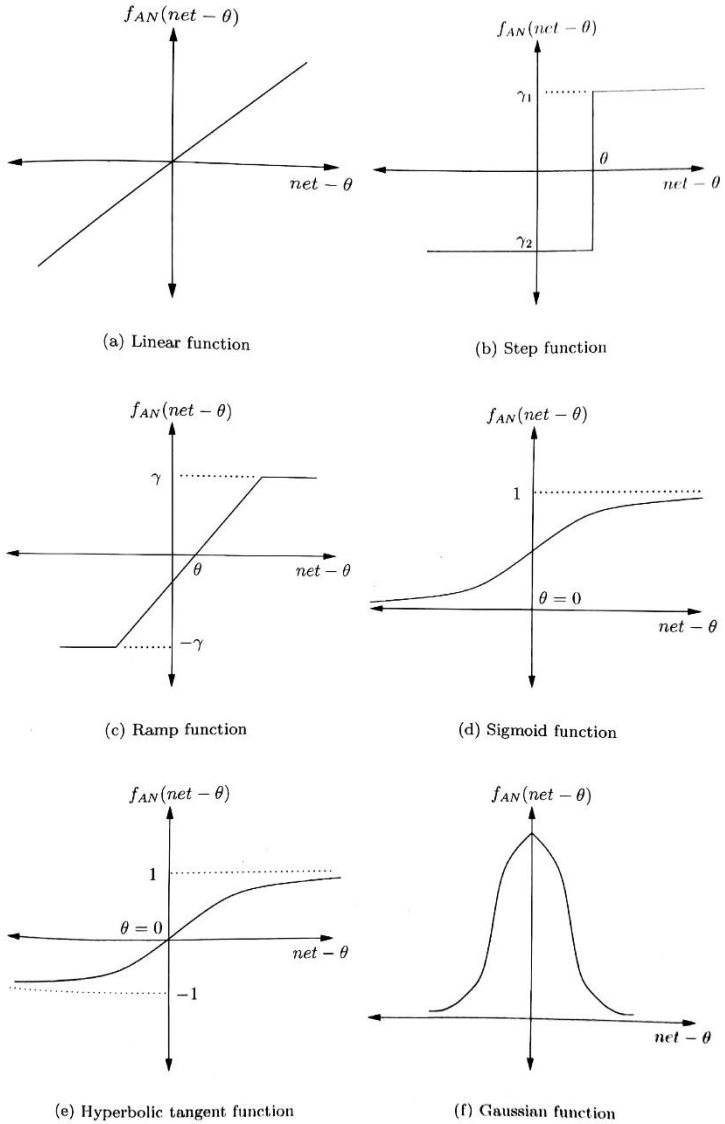


x_i – Entradas

w_i – Pesos

o – Salida

$$net = \sum_{i=1}^{n} w_i x_i$$



Redes Neuronales Artificiales (ANN)

Mapeo no lineal: $f_{ANN}: \mathbb{R}^n \rightarrow \mathbb{R}^k$

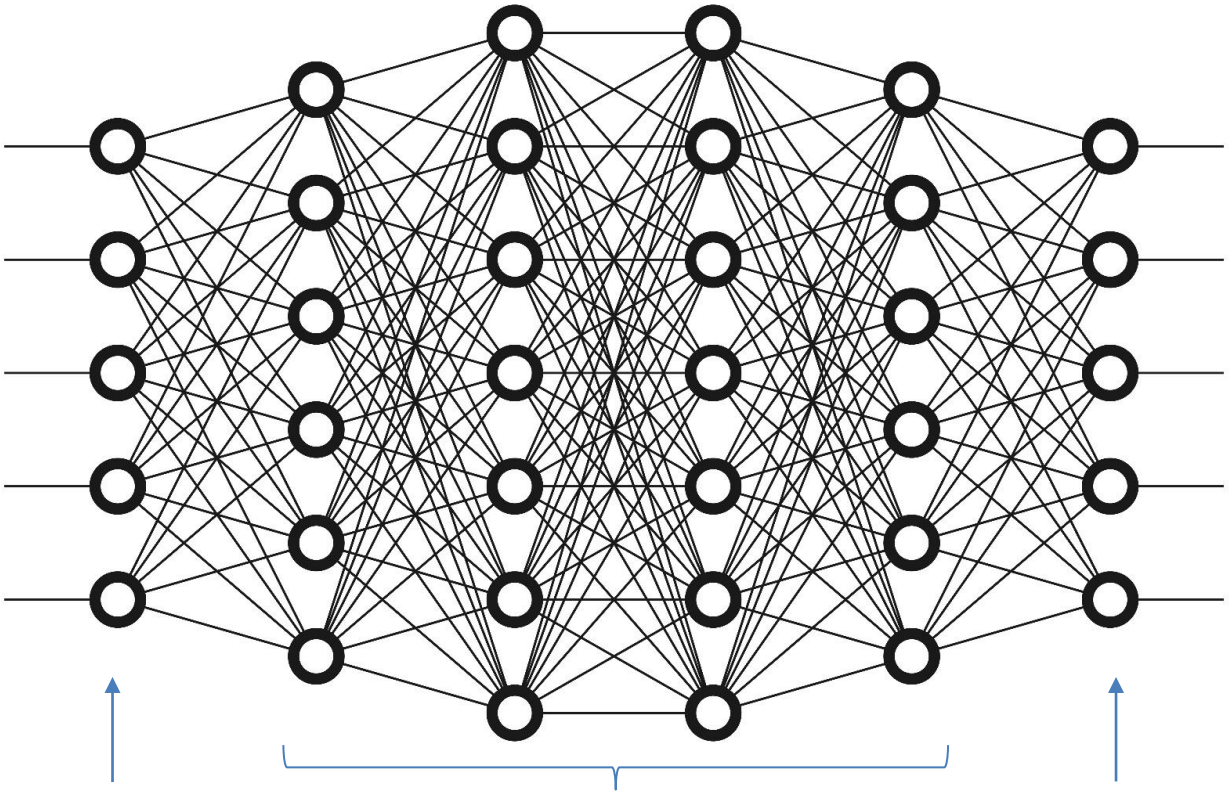
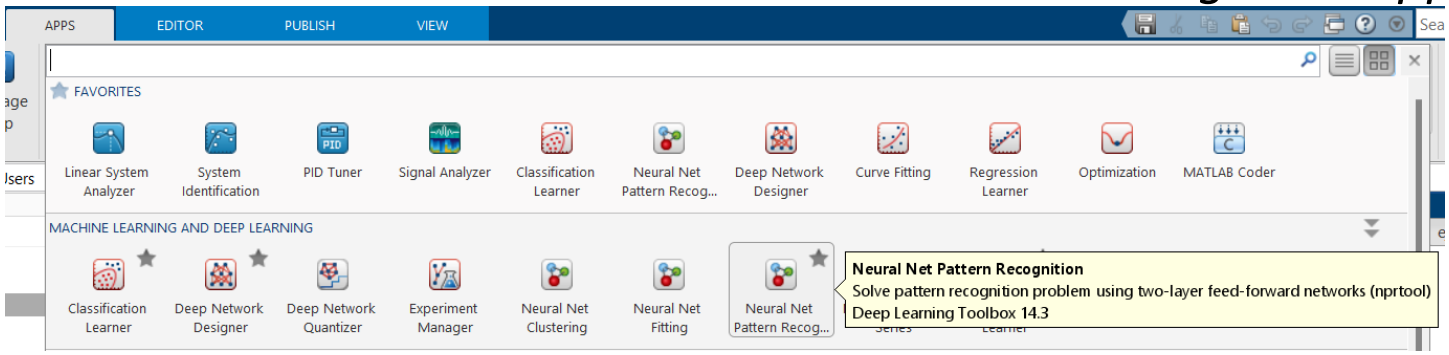
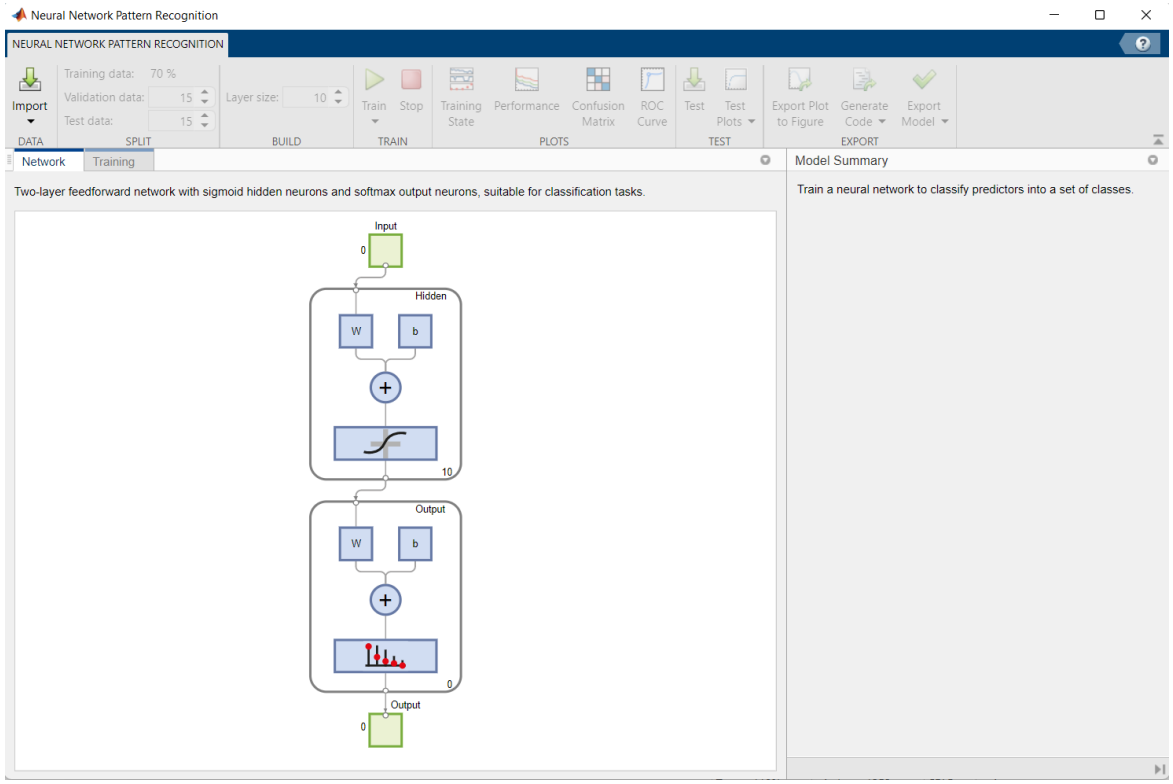


Figura tomada de: https://iq.opengenus.org/artificial-neural-networks/

Redes Neuronales Artificiales (ANN)

Veamos cómo se usa el *Neural Network Pattern Recognition app* de Matlab.





Script de ejemplo:
ej_ann.m

Matriz de Confusión

Es una forma conveniente de visualizar el desempeño de un algoritmo/método de clasificación.

Cada columna muestra el número (o porcentaje) de predicciones de cada clase (i.e., las muestras que el clasificador asignó a cada clase), y cada fila representa las instancias en las clases reales (algunos autores invierten filas y columnas; no importa realmente).

		Clase Asignada			
		ω_1	ω_2	$\cdots$	ω_n
Clase Real	ω_1	c_{11}	c_{12}	$\cdots$	c_{1n}
	ω_2	c_{21}	c_{22}	$\cdots$	c_{2n}
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	ω_n	c_{n1}	c_{n2}	$\cdots$	c_{nn}

c_{ij} es el número (o porcentaje) de muestras pertenecientes a la clase i que fueron clasificadas como clase j.

Los elementos de la diagonal son los que fueron clasificados correctamente.

Matriz de Confusión

Matriz de Confusión con Verdadero/Falso o Positivo/Negativo (caso particular: 2 clases).

Por ejemplo, un test de COVID-19.

		Diagnóstico	
		P	N
Condición Real	d.p. /c.p.	VP	FN
	d.n. /c.a	FP	VN

VP = Verdadero Positivo (diagnóstico positivo, condición presente).

VN = Verdadero Negativo (diagnóstico negativo, condición ausente).

FP = Falso Positivo (diagnóstico positivo, condición ausente).

FN = Falso Negativo (diagnóstico negativo, condición presente).

Acerca de la Dimensionalidad

En aplicaciones prácticas, es común encontrar problemas involucrando decenas o cientos de características (*features*) (ej. imágenes).

Aspectos importantes:

- ¿Cómo afecta la dimensionalidad y la cantidad de datos de entrenamiento al rendimiento de los algoritmos de clasificación?
- ¿Cuál es la complejidad computacional del clasificador?

Acerca de la Dimensionalidad

Algunas consideraciones:

- Cada característica contribuye a disminuir la probabilidad de error. Agregar más características puede ayudar a disminuir el error en la clasificación.
- Muchas características pueden llevar a cálculos/cómputos inviables.
- Mientras más características se usen, se requieren muchas más muestras para obtener resultados significativos.
- "*Curse of dimensionality*"

Ejercicios (1)

1. Entrenar un SVM, clasificar muestras de prueba, y construir la correspondiente matriz de confusión. **Entrega:** código y matriz.
2. Entrenar una ANN, clasificar muestras de prueba, y construir la correspondiente matriz de confusión. **Entrega:** Imagen de la red y matriz.

Deberá subir un pdf a Google Drive con los resultados, código, gráficas, imágenes y respuestas de estos ejercicios y los de aprendizaje no supervisado.

Aprendizaje No Supervisado

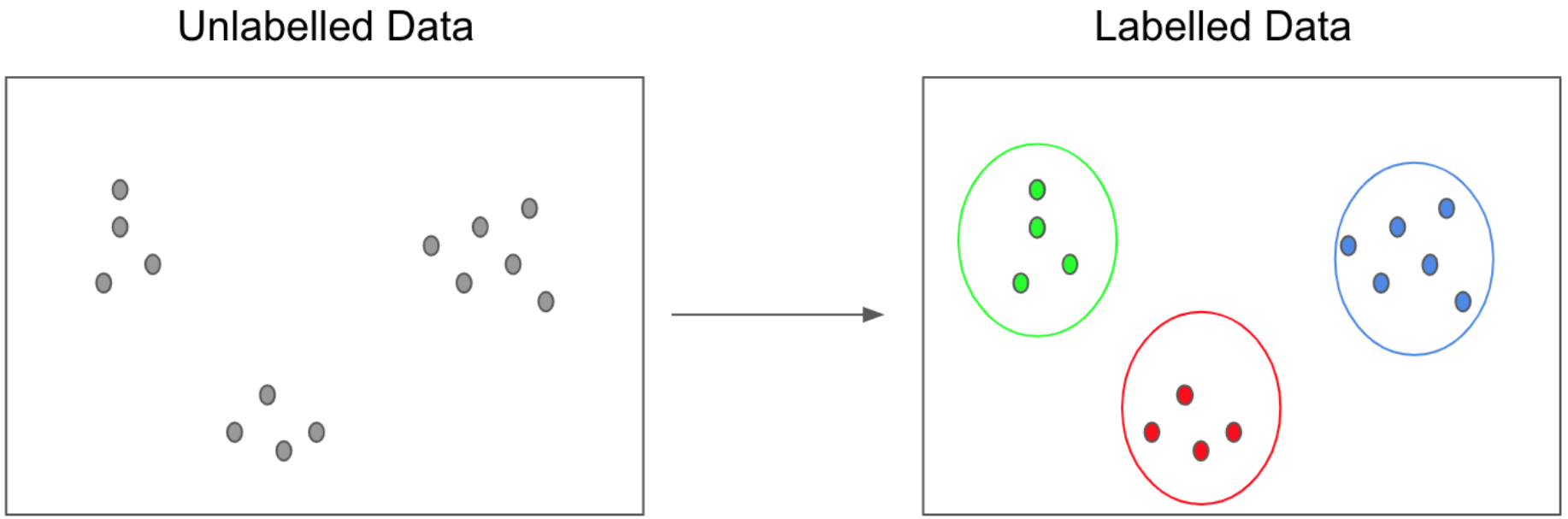


Figura tomada de: https://datascience.eu/machine-learning/clustering-algorithms-and-their-significance-in-machine-learning/

Aprendizaje No Supervisado y Agrupamiento

La diferencia fundamental con el aprendizaje supervisado es que no se conocerán las etiquetas de los datos o vectores de muestra de entrenamiento.

El enfoque principal es el de agrupar las muestras en conjuntos/grupos/cúmulos (*clusters*) "sensatos" (significativos). Esto permitirá descubrir similitudes y diferencias entre las muestras, y llegar a conclusiones útiles sobre las mismas.

El procedimiento de agrupar muestras se conoce como **Agrupamiento** o ***Clustering***.

Ejemplo

Considere los siguientes animales:

cat	dog	frog	goldfish	lizard	red mullet
seagull	shark	sheep	sparrow	viper	

¿Cómo podemos agruparlos?

Para ello, necesitamos definir algún **criterio de agrupamiento**.

Ejemplo

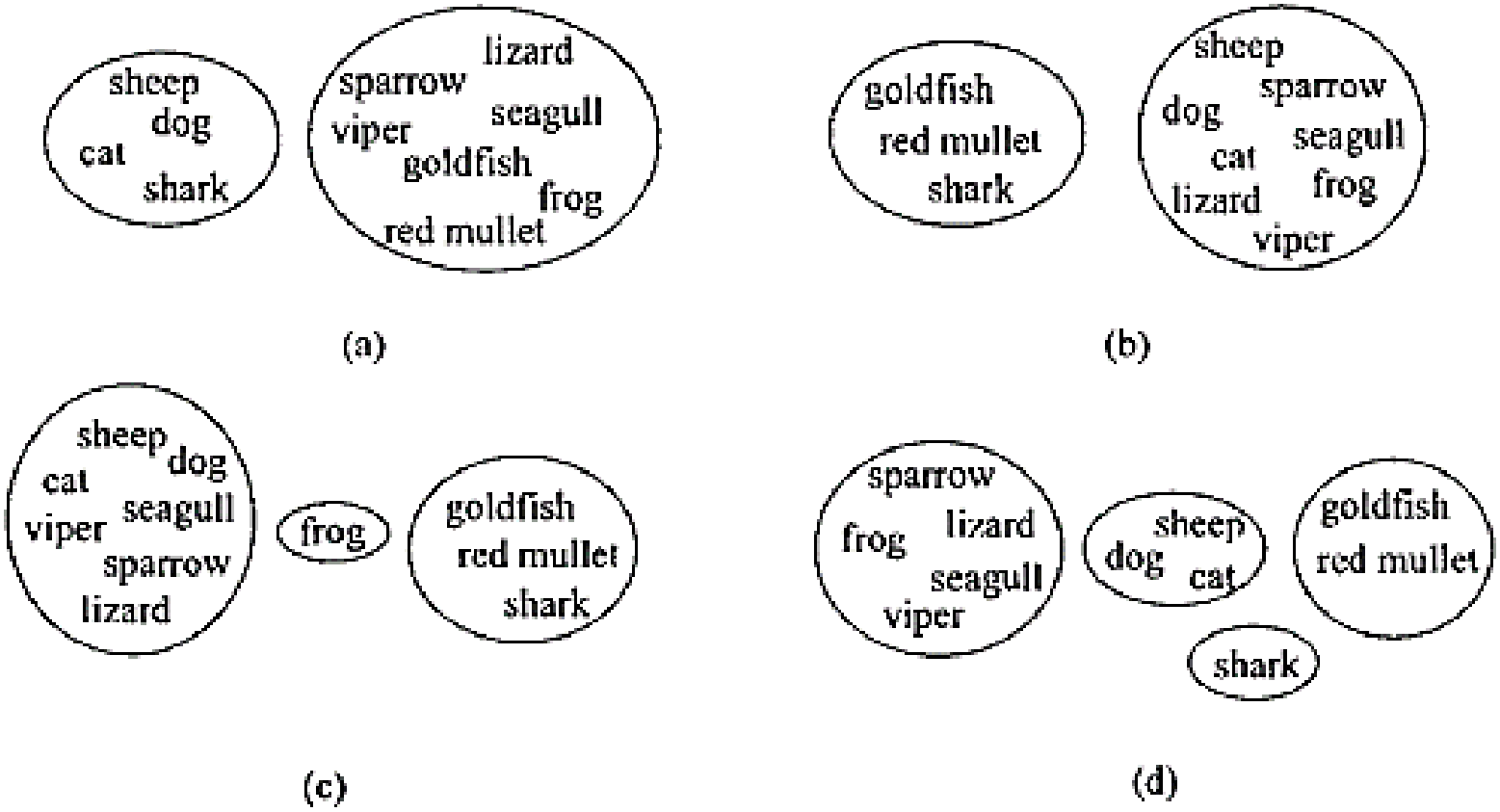


FIGURE 11.1: Resulting clusters if the clustering criterion is (a) the way the animals bear their progeny, (b) the existence of lungs, (c) the environment where the animals live, and (d) the way these animals bear their progeny and the existence of lungs.

[7]

Clustering

Dependiendo del criterio usado, los resultados del agrupamiento pueden ser muy variados.

Los humanos hacemos agrupamiento de la información que recibimos constantemente. Procesar cada entidad que percibimos de forma individual sería imposible. Tendemos a categorizar las entidades en grupos.

Clustering

Como en el caso de aprendizaje supervisado, podemos asumir que los patrones o muestras son representadas en términos de características (*features*), que forman vectores de características $d-$ dimensionales.

Aspectos importantes a considerar son:

- Selección de características
- Medidas de proximidad: para cuantificar qué tan similares o distintos son los vectores de características entre sí.
- Criterio de agrupamiento
- Algoritmos de agrupamiento
- Validación de los resultados
- Interpretación de los resultados

Clustering

Considere las muestras siguientes. ¿Cuántas formas “sensatas” de agrupar los datos hay?

Clustering

¿Cuál es "correcta"? ¿Hay otras opciones? Inexorablemente, debemos lidiar con la subjetividad.

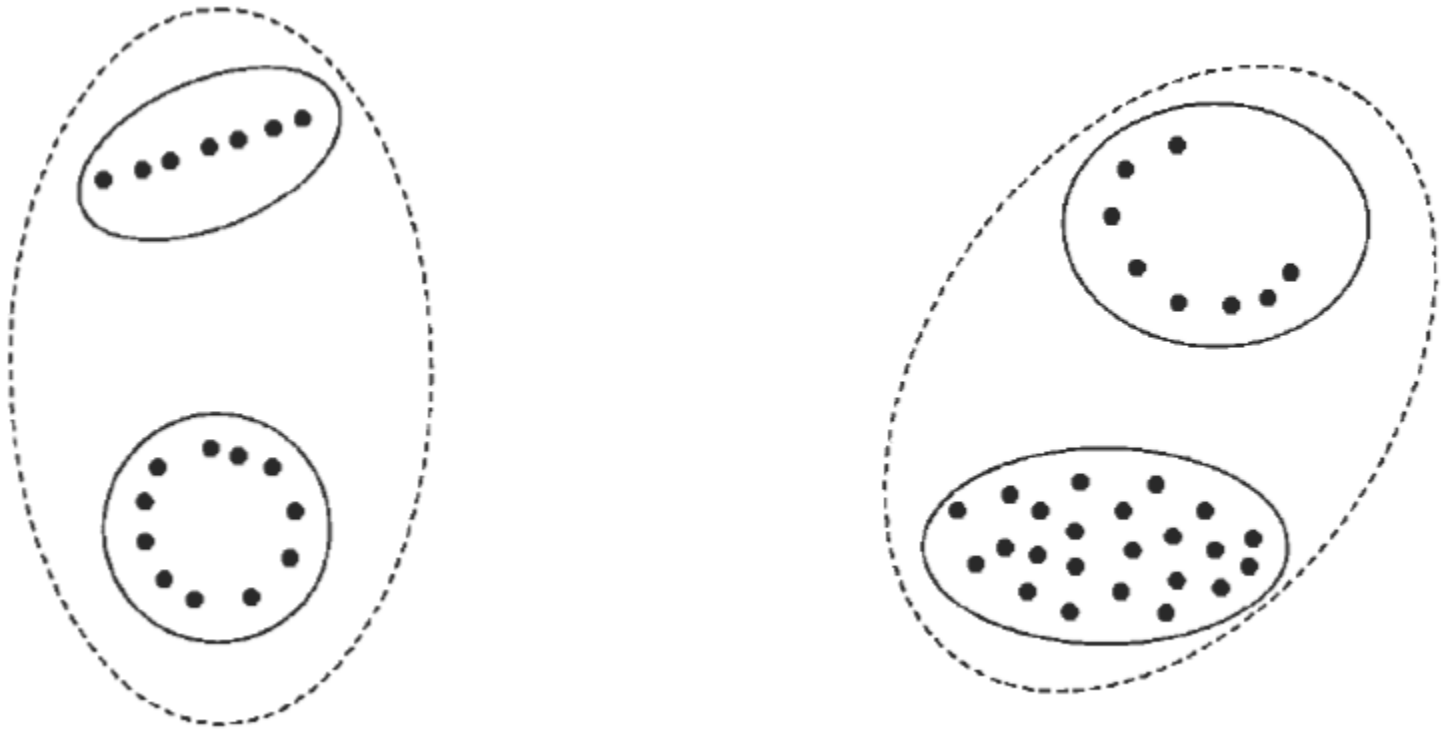


FIGURE 11.2: A coarse clustering of the data results in two clusters, whereas a finer one results in four clusters. [7]

Algunas Definiciones de *Clustering*

Sea X un conjunto de N datos (muestras).

$$X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$$

m-clustering de X: es la partición de X en m conjuntos (*clusters*), $C_1, \dots, C_m$, tal que se cumplen las siguientes condiciones:

1. $C_i \neq \emptyset, \quad i = 1, \dots, m$
2. $\bigcup_{i=1}^{m} C_i = X$
3. $C_i \cap C_j = \emptyset, \quad i \neq j, \quad i, j = 1, \dots, m$

Algunas Definiciones de *Clustering*

Los vectores contenidos en el *cluster* C_i son "más similares" entre sí, y "menos similares" a los vectores de otros *clusters*.

Cuantificar el término "similar" depende de los tipos de *clusters* involucrados. Distintas métricas se requerirían para los *clusters* mostrados en la siguiente figura.

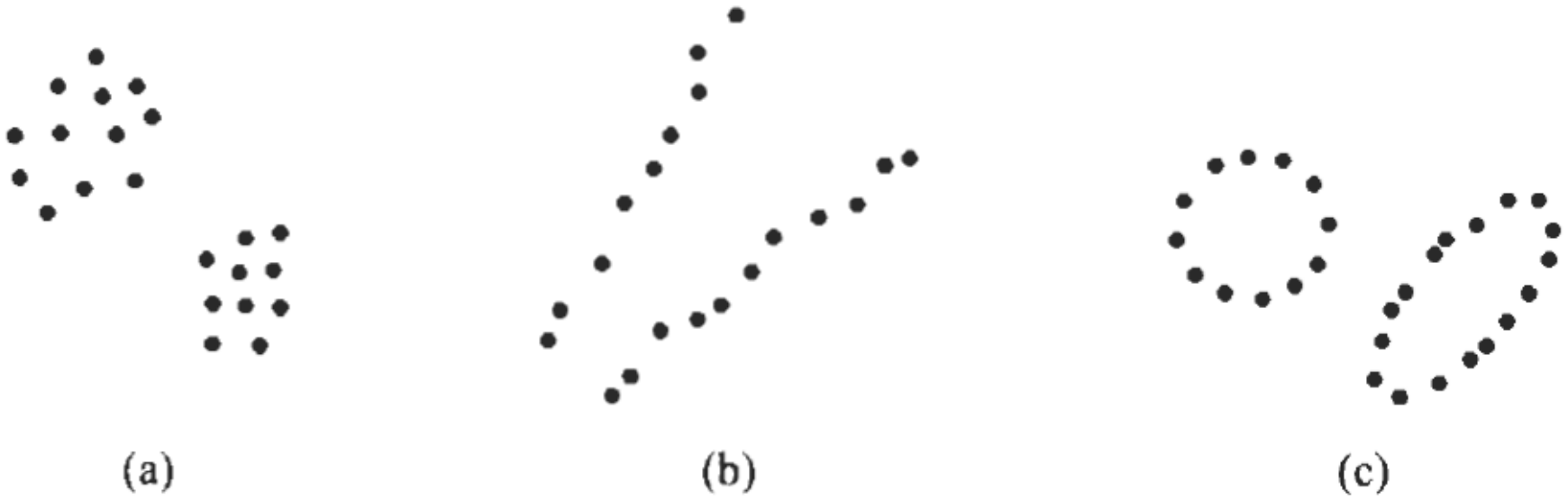


FIGURE 11.3: (a) Compact clusters. (b) Elongated clusters. (c) Spherical and ellipsoidal clusters.

[7]

Algunas Definiciones de *Clustering*

En la definición anterior, cada vector pertenece a un único *cluster*. Esto se conoce como agrupamiento duro (*hard* o *crisp clustering*).

Existe el *soft clustering* (*fuzzy clustering*).

Algoritmos de *Clustering* Comunes

- k-Means (c-Means, Isodata)
- Jerárquicos
- Fuzzy c-Means

k-Means (c-Means, Isodata)

- Ejemplo de agrupamiento duro. Es, tal vez, el más popular de todos los algoritmos de agrupamiento.
- Usa puntos como representantes de los *clusters*, y el cuadrado de la distancia Euclidiana como medida de disimilitud.
- Es computacionalmente simple.
- Es conveniente para recuperar *clusters* compactos.

- Se necesita saber el número de *clusters*.
- Es sensible a la inicialización de los centros de los *clusters*.
- Es sensible a los datos atípicos (*outliers*). Los centros se pueden mover hacia esos datos, desviándose de la posición "correcta".

Ejemplo en Matlab:
ej_k_means.m

Ejercicios (2)

1. Descargue el archivo **k_means_datos.mat**. Verá dos variables: X y X_out.
2. Aplique k_means a las muestras X. Pruebe varios valores de m, y corra varias veces el algoritmo para cada m. ¿Observa siempre lo mismo? ¿Nota algún problema?
   Entrega: a) Dos imágenes de los resultados, una correspondiente a la m que más le hace sentido, y otra correspondiente a otra m. b) Las respuestas.
3. Ahora aplique k_means a las muestras X_out. Pruebe varios valores de m, y corra varias veces el algoritmo para cada m. ¿Qué observa?
   Entrega: a) Una imagen de los resultados usando la m que más le hace sentido. b) La respuesta.

Deberá subir un pdf a Google Drive con los resultados, código, gráficas, imágenes y respuestas de estos ejercicios y los de aprendizaje supervisado.

Nota: El *"Statistics and Machine Learning toolbox"* de Matlab cuenta con una función **kmeans** y otras funciones para distintos tipos de *clustering*, así como funciones y apps para entrenar y aplicar clasificadores, hacer regresiones, etc.

Referencias

1. https://supplychainbeyond.com/what-is-machine-learning/
2. https://www.wordstream.com/blog/ws/2017/07/28/machine-learning-applications
3. http://librfidhub.blogspot.com/2013/09/biometrics_13.html
4. http://vigir.ee.missouri.edu/index.html
5. http://vigir.missouri.edu/~lulorivera/EMGWheelchair.htm
6. Duda, R., Hart, P. and Stork, D. G. (2001). Pattern Classification. Wiley, 2nd ed.
7. Koutroumbas, K. and Theodoridis, S. (2008). Pattern Recognition. Academic Press, 4th ed.
8. https://deepai.org/machine-learning-glossary-and-terms/curse-of-dimensionality
9. https://builtin.com/data-science/curse-dimensionality