

Clasificación de Imágenes por objetos y redes neuronales

Dr. Rodrigo López Farías

Unidad 1. Introducción a las técnicas de Clasificación

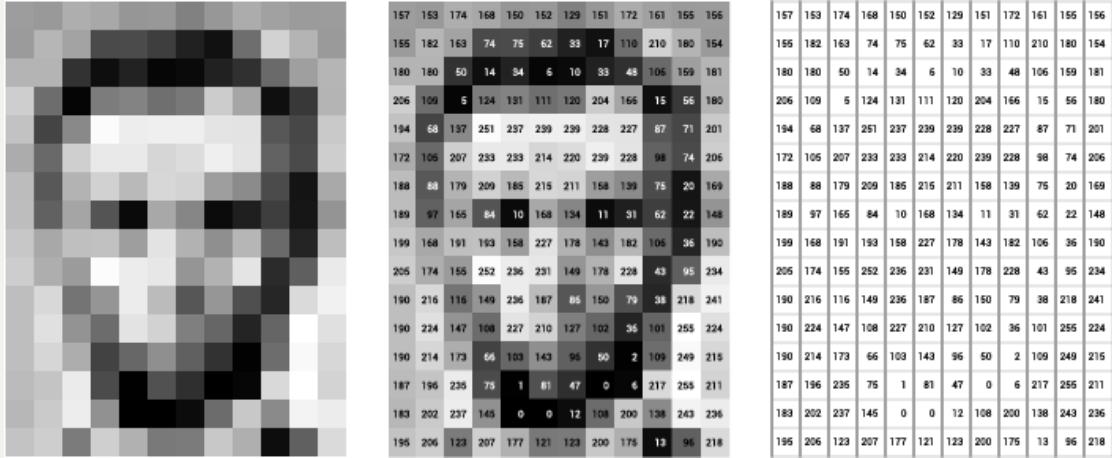
1.0 Introducción al aprendizaje automático.

- En los primeros días de la inteligencia artificial, este campo tomó y resolvió los primeros problemas que intelectualmente son difíciles para los humanos pero muy fáciles para las computadoras.
- Se ha cuestionado que es inteligencia Artificial. Si no es inteligencia humana que es?
 - Deep y deeper Blue que derrotaron al ajedrecista Gary Kaspárov en 1996. Es una Inteligencia Artificial? 

<https://www.professional-ai.com/deep-blue-algorithm.html>

Retos de la inteligencia Artificial

- Resolver problemas fáciles e intuitivos pero que son difíciles de describir formalmente para las personas.
- Por lo tanto parte de este problema consiste en resolver como el conocimiento informal debe ser representado en una computadora.



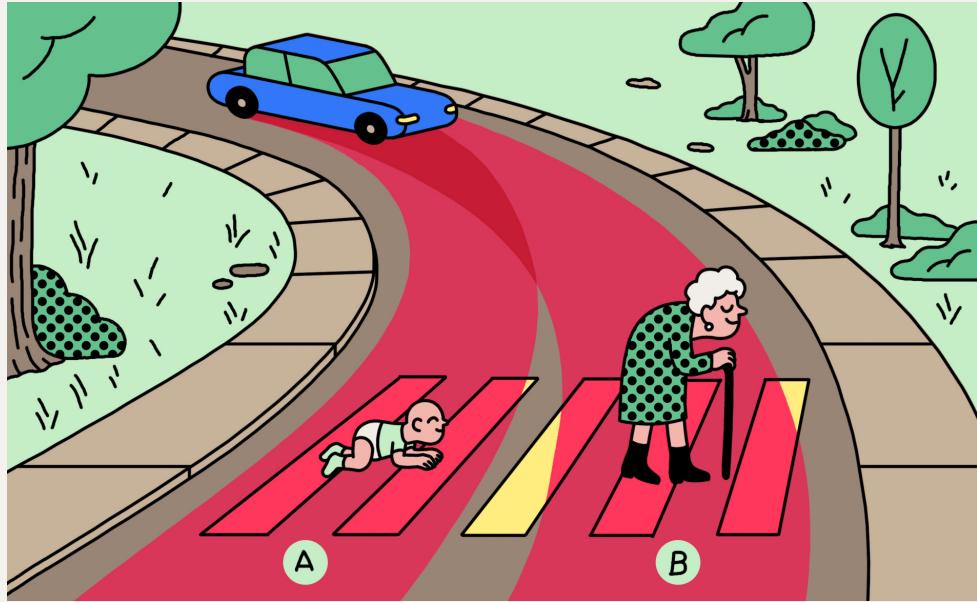
Representación de una Imagen en una Matriz.

Ejemplos de problemas que son fáciles para los humanos pero difíciles para las computadoras.

- Entender el lenguaje natural.
- Inferir el sentimiento de las personas.
- Traducción Automática
- Clasificación de Imágenes.
- Reconocimiento de objetos.
- Qué otros problemas los humanos pueden resolver fácilmente?
 - La Justicia

Hay otros problemas que van mas allá de tema pero vale la pena mencionar son, aquellos problemas de decisión difíciles para las computadoras y para los humanos. El problema es si los humanos delegamos a una computadora tomar decisiones éticas.

- Decisiones éticas.



<https://www.technologyreview.com/2018/10/24/139313/a-global-ethics-study-aims-to-help-ai-solve-the-self-driving-trolley-problem/>

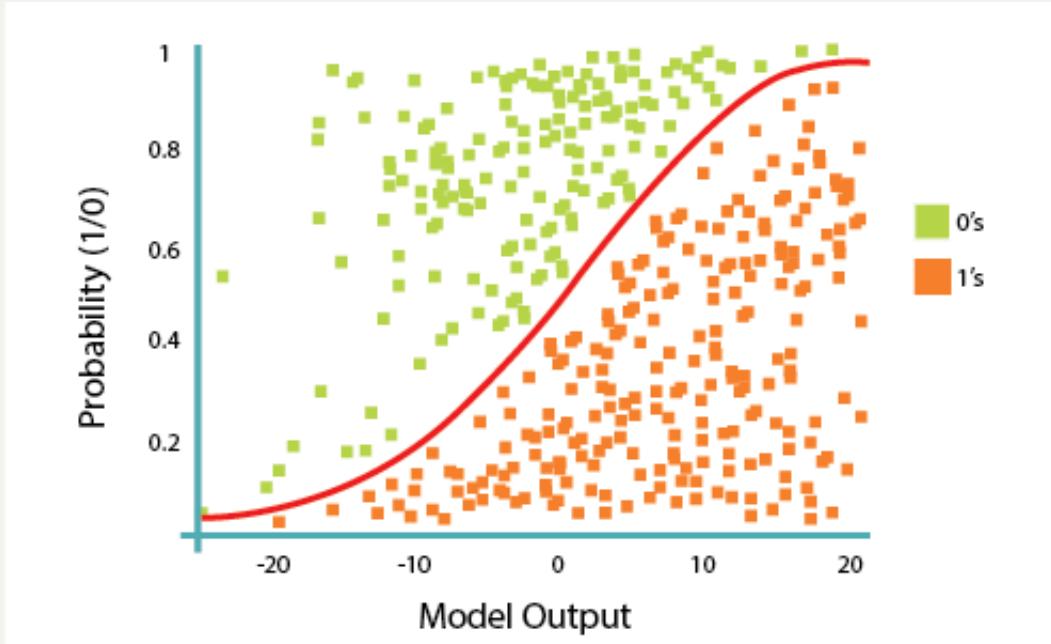
Por lo tanto, el Aprendizaje automático consiste en dar solución a estos problemas intuitivos aprendiendo de la experiencia.

En aprendizaje profundo, la idea principal es entender la información como una jerarquía de conceptos complicados a partir de otros conceptos mas simples.

En los inicios se intentó crear inteligencia artificial con codificación "dura". (Introducción explícita de reglas de inferencia en una base de datos llamado Enfoque de **Conocimiento Base** para la inteligencia artificial)

La rigidez de este tipo de enfoque, sugiere que los sistemas de IA, deben tener la habilidad de adquirir su propio conocimiento, extrayendo patrones de los datos en bruto. Esta capacidad de las computadoras se le llama Aprendizaje Automático (AA) o "**Machine Learning**".

AA pudo hacer que las computadoras aborden problemas complejos para los humanos utilizando el conocimiento del mundo real para tomar decisiones que parecen subjetivas. Por ejemplo, uno de los primeros éxitos es la implementación de una regresión logística simple que es capaz de recomendar o no parto por cesarea a los médicos. (Principio utilizado por las redes neuronales)



El truco para que funcionen estos modelos reside en **una buena representación de los datos** que incluye una **selección de características** cuidadosa por un experto.

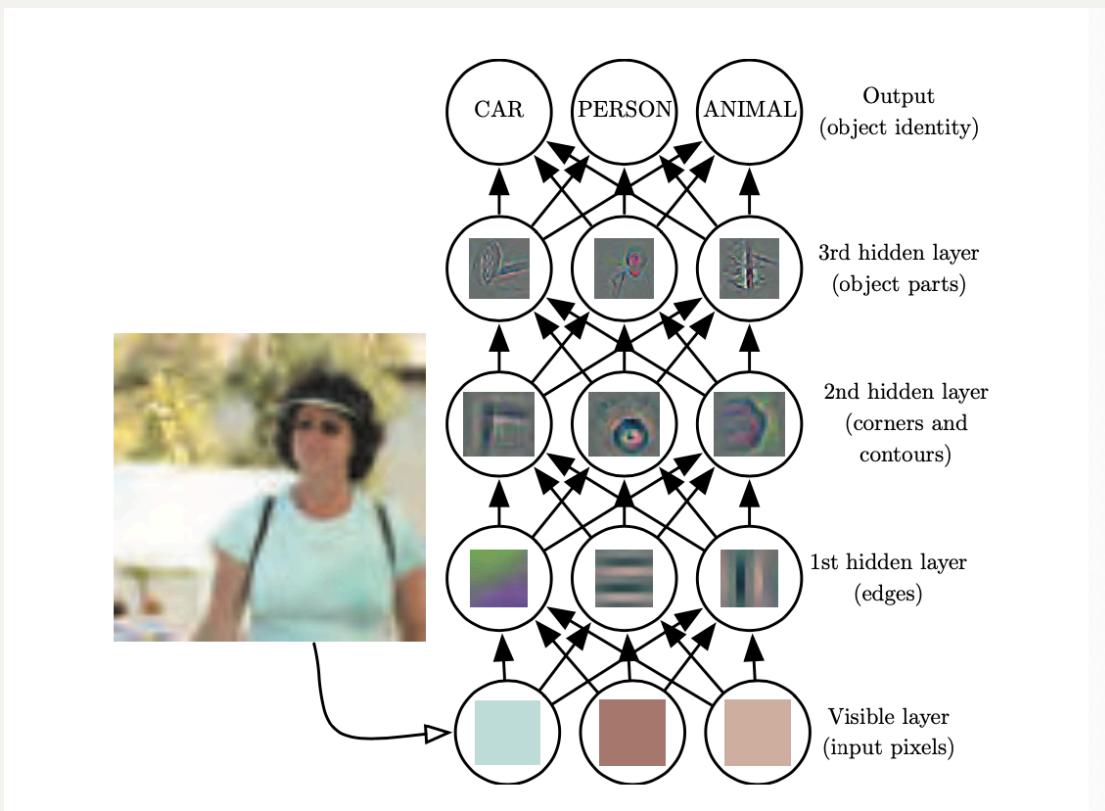
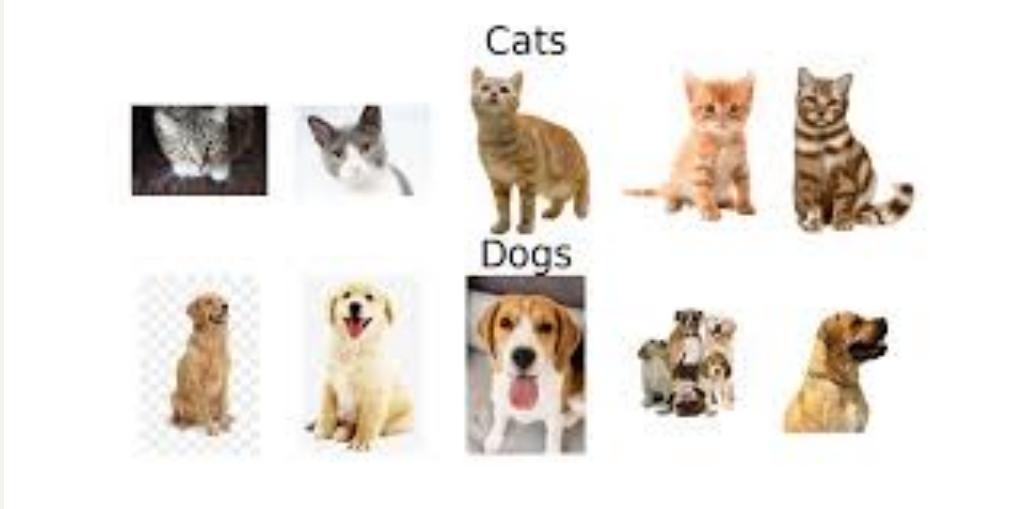
Una mala selección o representación de datos, genera malos algoritmos predictivos.

Para muchas tareas del mundo real es muy complicado distinguir las características que deben ser utilizadas para entrenar un modelo.

Por ejemplo, como distinguir una persona de un carro, o de un perro.

Para esto, una solución es utilizar **AA para aprender automáticamente la representación misma de los datos**.

Cuando se diseñan algoritmos para aprender características, una de las metas es distinguir los factores invariantes o características de los factores de variación o ruido que explican los datos observados.



ML responde a la pregunta de cual es el mejor modelo para explicar datos.

Ejemplo de como el aprendizaje profundo extrae características importantes

Que características invariantes representan a un gato o a un perro?

Machine Learning es una subárea de la IA que detecta patrones de manera autoática en los datos

Antecedentes

Los modelos predecesores, están basados en modelos lineales simples, que trataron de asociar (o mapear) una entrada de tamaño n $\mathbf{x} = \{x_1, x_2, \dots, x_i, \dots, x_n\}$ a una salida y_i .

Estos modelos aprenden un conjunto de pesos $\mathbf{w} = \{w_1, w_2, \dots, w_n\}$, para ajustar una función f :

$$f(\mathbf{x}, \mathbf{w}) = x_1 w_1 + x_2 w_2 + \dots + x_n w_n$$

tal que la suma de los errores $\sum(f(\mathbf{x}_i, \mathbf{w}) - y_i)$ sea la mínimo.

Ecuación de la recta

$$y_i = \mathbf{w} \mathbf{x}_i + b$$

Primera Tarea: 1 ejemplo de aplicación de la ecuación de la recta.

Ola Cero

(McCulloch and Pitts, 1943) Utiliza el modelo lineal simple

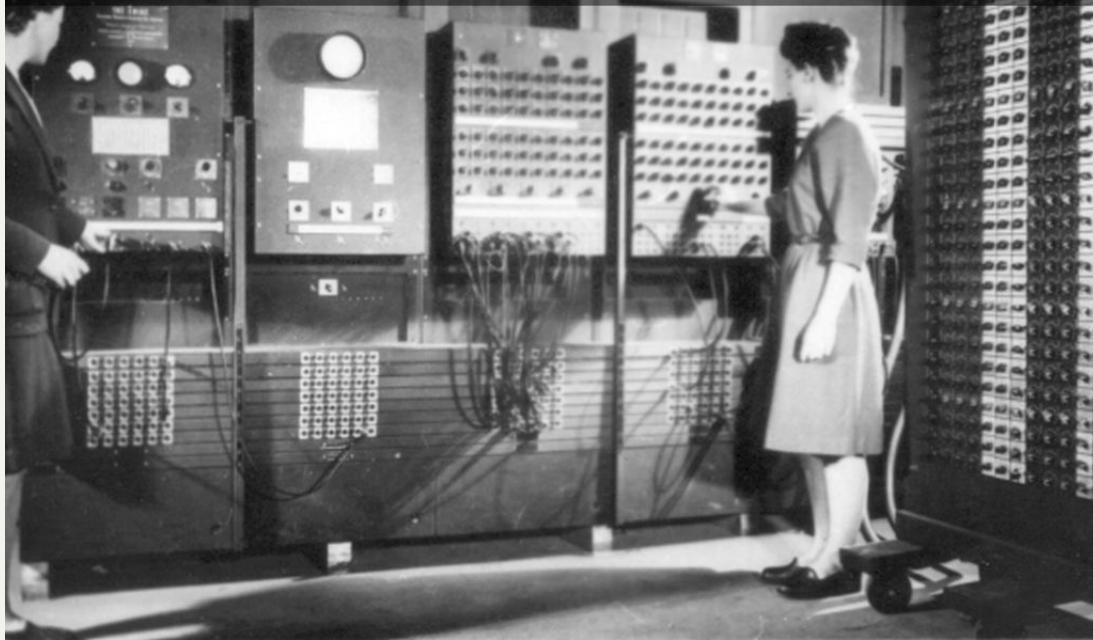
$$\text{sign}(f(\mathbf{x}, \mathbf{w})) \in \{-1, 1\}$$

Este modelo puede reconocer dos categorías. Los pesos \mathbf{w} son ajustados manualmente.

No existe un mecanismo de entrenamiento.

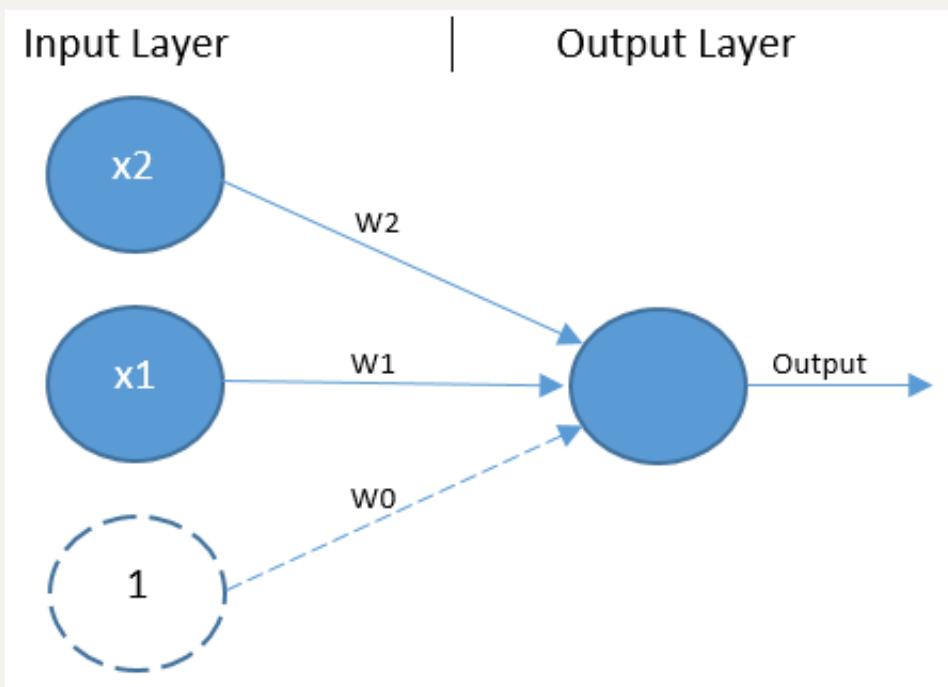
En ese tiempo sale la primera computadora digital de propósito general: ENIAC

ENIAC: 1943-46



Primera ola:

Cibernética (1940-1960, Rosenblatt 1958). Impulsado por la teoría de aprendizaje Biológico. Nace el Modelo Perceptrón., que es el primer modelo de una sola neurona capaz de aprender de los datos.



<https://medium.com/@jayeshbahire/the-xor-problem-in-neural-networks-50006411840b>

(Minsky and Papert, 1969) Estos modelos lineales tienen limitaciones. El mas famoso es la incapacidad de resolver la función XOR.

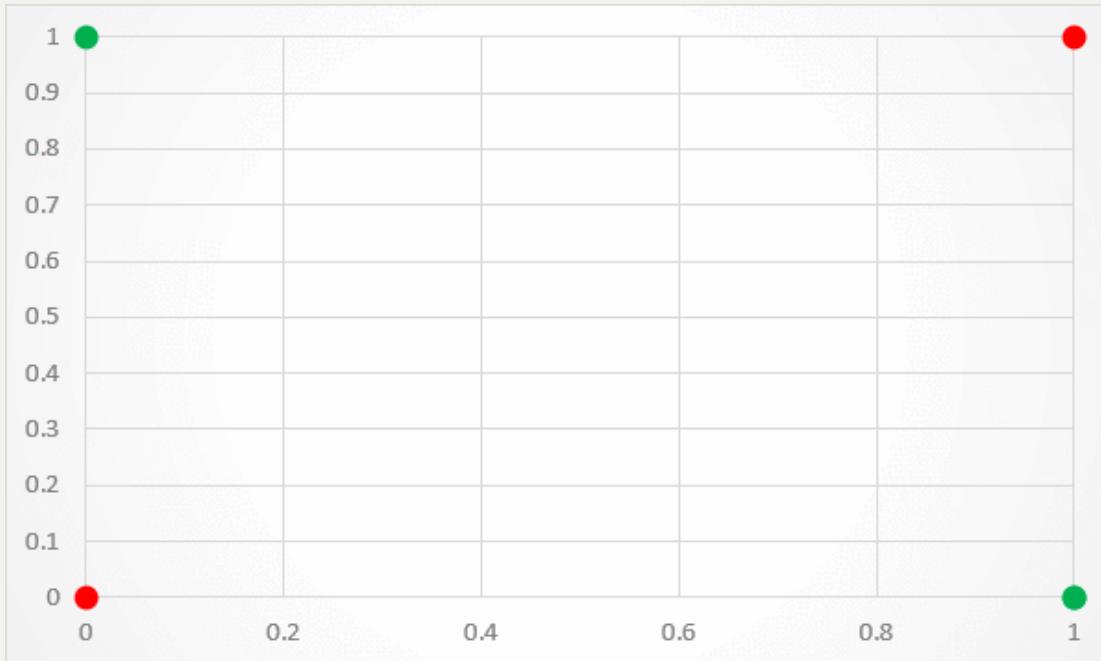


Tabla de verdad de la operación lógica xor

x | xor | y

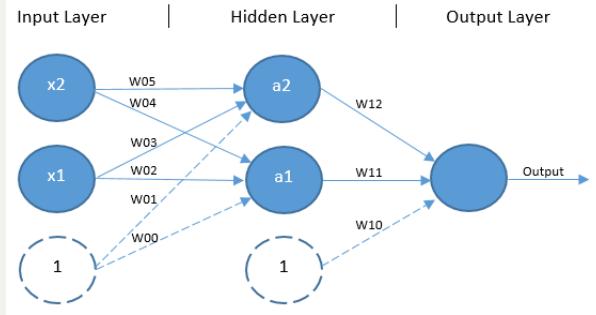
0 | xor | 0 = 0

0 | xor | 1 = 1

1 | xor|1 = 0

1 |xor | 0 = 1

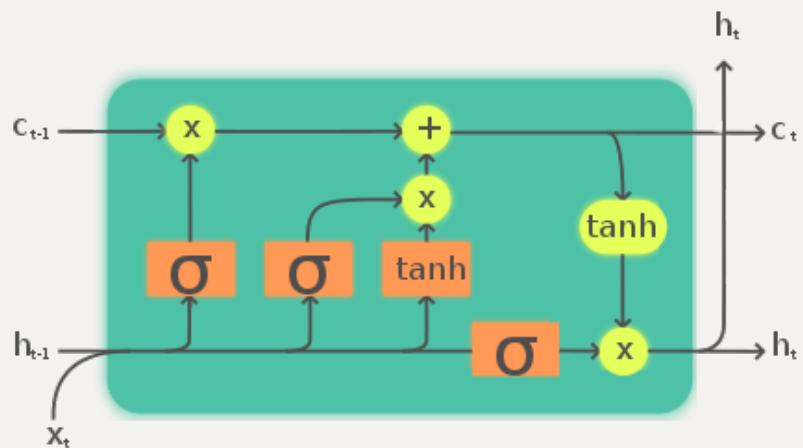
Segunda ola: Conexionismo: (1980-1995, Rumelhart) Impulsado por una red neuronal con una o dos capas ocultas que resuelve el problema del XOR agregando una capa oculta.



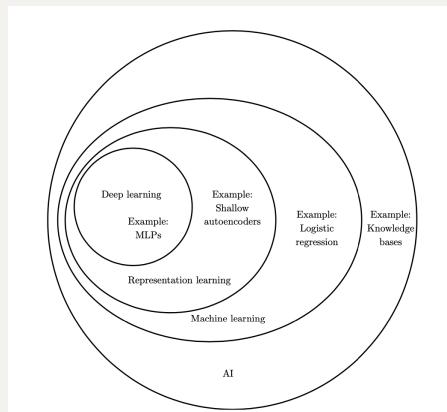
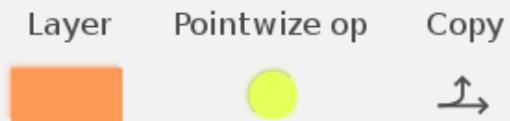
Tercera ola: 1995 - Deep learning. Impulsado por Redes Neuronales Multi-Capa y algoritmo De Gradiente Estocástico Descendente.

Estructura de la red	Tipo de región de decisión	Solución OR Exclusivo	clases con regiones geométricas	Formas de superficie de decisión más generales
	Hiperplano simple			
	Regiones convexas abiertas o cerradas			
	Arbitrario (complejidad limitada por el numero de nodos)			

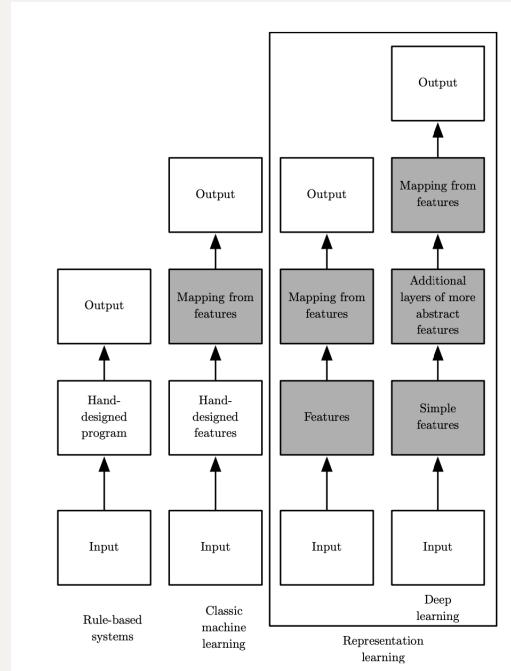
Hochreiter (1991) y Bengio et al. (1994), Se identifico otro problema con el aprendizaje profundo que eses el modelado de secuencias muy largas, para esto Hochreiter y Schmidhuber (1997) propusieron the long short-term memory (LSTM) para resolver estos problemas. LTSM es una red neuronal recurrente que incorpora una operación para definir el que tanto importa recordar la información para hacer una Clasificacióno regresión.



Legend:



Deep Learning son un tipo de aprendizaje automático que, ademas de encontrar una función de mapeo para asociar un conjunto de características x con una salida datos (y), internamente logran aprender como representar esos datos.



Técnicas modernas de AP, toman ideas de muchas otras áreas, como

- Álgebra Lineal
- Probabilidad
- Teoría de la información
- Optimización Numérica

Tipos de Aprendizaje automático

Tomando como guía la figura del diagrama de Venn, y enfocándonos en el área de Aprendizaje Automático, podemos identificar dos grandes tipos de aprendizaje.

1) Aprendizaje Supervisado

El aprendizaje supervisado trata de aprender una función de mapeo aproximada $\hat{f} : \mathbf{x}_i \rightarrow \hat{y}_i$ a la función real $f : \mathbf{x}_i \rightarrow y_i$ dado a un conjunto de pares de datos etiquetados

$$D = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$$

donde:

- D : es el conjunto de entrenamiento.
- N : número de muestras
- \mathbf{x}_i : Atributos o características, pueden ser discretas, continuas o reales.
- y_i : Variable Categórica con valores (**Clasificación**) $y_i \in \{1, \dots, C\}$, o (**Regresión**) $y_i \in R^{Dim}$
- Si $C = 2$ significa que trata de un problema de **clasificación binaria**.
- Cuando $C > 2$, se le llama **clasificación Multi-clase**.
- Si un objeto puede pertenecer a diferentes clases al mismo tiempo se le llama **clasificación Multi-etiqueta**.
- La meta principal es ajustar una función de aproximación con capacidad de **generalización** para clasificar de manera adecuada los datos mas allá del conjunto de entrenamiento.

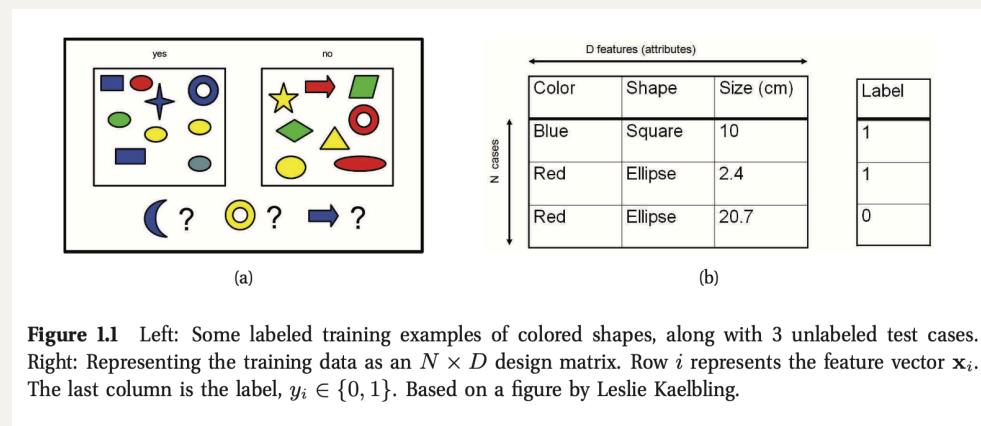


Figure 1.1 Left: Some labeled training examples of colored shapes, along with 3 unlabeled test cases. Right: Representing the training data as an $N \times D$ design matrix. Row i represents the feature vector \mathbf{x}_i . The last column is the label, $y_i \in \{0, 1\}$. Based on a figure by Leslie Kaelbling.

Ejemplo:

- Árboles de clasificación
- Clasificación por mínima distancia
- Redes neuronales.
- Modelos gausianos
- Modelos Bayesianos
- Maquina de Soporte de Vectores

2) Aprendizaje No supervisado

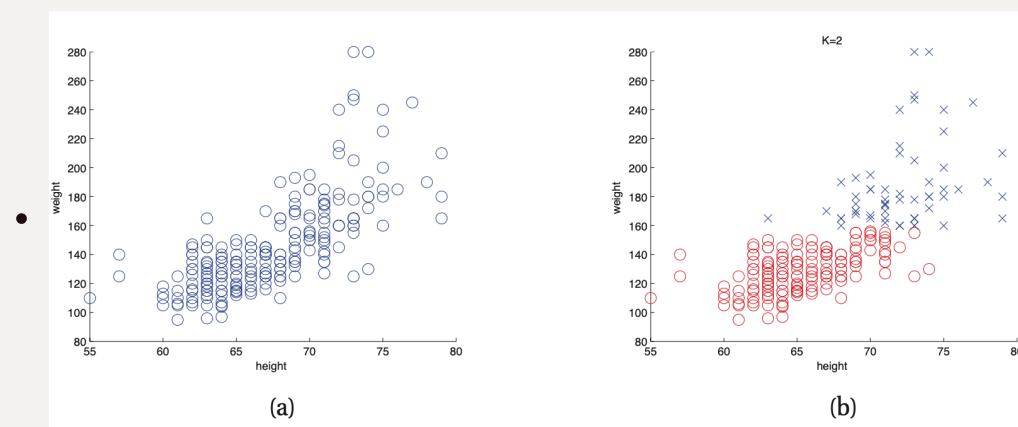
Este tipo de aprendizaje tiene un enfoque descriptivo con datos que no están previamente etiquetados. El conjunto de datos se representa como

$$D = \{\mathbf{x}_i\}_{i=1}^N.$$

La finalidad es encontrar la estructura general de los datos. Es de utilidad para

- Utilizado para etiquetado automático, cuando el etiquetado supervisado es imposible o muy costoso.
- Resuelve el problema de encontrar clúster o grupos de datos en los datos.
- Descubre factores latentes. (aquellas variables que dan mas información sobre el conjunto de datos).
- Estimación de densidades de distribución.

Aplicación de ejemplo: Descubrir cuantos grupos de datos existen en los datos. Lado Izquierdo, datos sin clasificar del peso y la altura de cada persona. En la derecha detección de dos grupos diferentes.



Modelos relacionados

- k-medias
- Mapas auto-organizados
- Dendogramas
-

Otros modelos no supervisados se pueden consultar en

https://www.cienciadedatos.net/documentos/37_clustering_y_heatmaps#Ejemplo_divisivo

Predictión Probabilista en la clasificación de los datos.

- La predicción probabilista nos permite manejar casos que no son claros para el humano.
- Es deseable que exista una función de probabilidad la cual para una entrada, arroje una probabilidad de pertenecer a una clase.
- La probabilidad de que un dato o vector de entrada pertenezca a una clase se puede expresar utilizando **probabilidad condicional**:

$$p(y|\mathbf{x}, D)$$

Donde D : es el conjunto de datos, \mathbf{x} es el vector de entrada y la función regresa un valor entre 0 y 1.

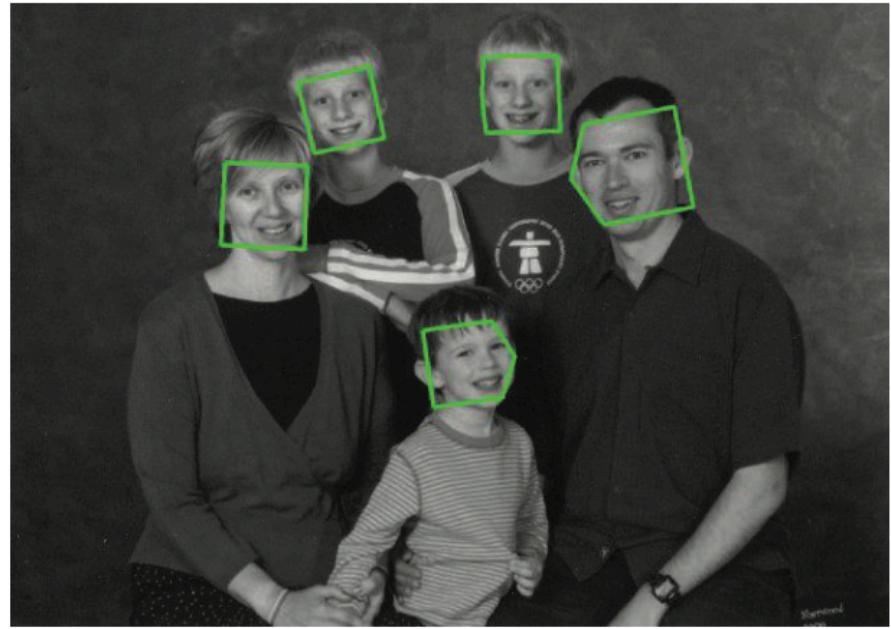
Para seleccionar la mejor estimación se utiliza un estimador de **Maximum a posteriori**

$$\hat{y} = \hat{f}(\mathbf{x}) = \operatorname{argmax}_c p(y = c | \mathbf{x}, D)$$

donde $c \in \{1, \dots, C\}$

Aplicaciones

- Clasificación de Documentos y filtro de spam. (La probabilidad de que un documento pertenezca a spam)
- Clasificador de flores.
- Clasificación de imágenes
- Reconocimiento de escritura
- Encontrar objetos dentro de una imagen con ventana deslizante (detección y localización de rostros).



- Predicción de series de tiempo
- Predecir la edad de un espectador dado un video de YouTube.
- Predecir la temperatura dentro de una locación, utilizando datos atmosféricos, hora del día, materiales de construcción, sensores etc...
- Predicción de concentraciones de contaminantes.

Descubrimiento de agrupamientos:

El descubrimiento de agrupamiento (o clustering tiene dos objetivos)

- Objetivo 1. Estimar la distribución de probabilidad de un conjunto de datos D con K grupos. Esto se puede expresar como la probabilidad de tener K grupos dado un conjunto de datos D

$$p(K|D)$$

Para encontrar K , se requiere una manera de medir el ajuste de los datos los diferentes K grupos, hasta maximizar la probabilidad de que el conjunto de datos D sea generado por una distribucion de probabilidad.

$$K^* = \operatorname{argmax}_K p(K|D)$$

Objetivo 2: Estimar a que grupo pertenece un dato.

Por ejemplo haciendo $z_i \in \{1, \dots, K\}$, donde cada punto o dato i , es asignado a una de esas clases.

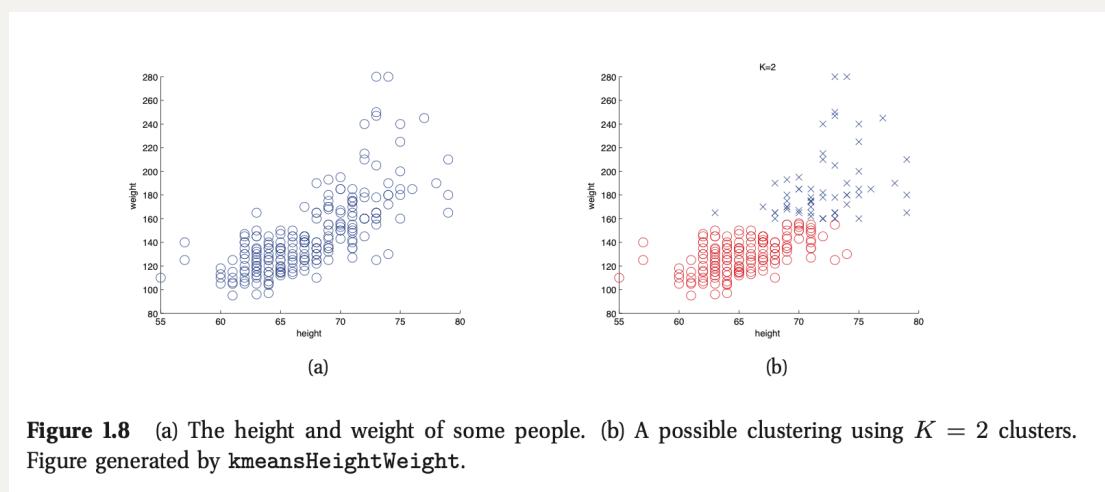
Como z_i es una variable calculada y no existe en los datos de entrenamiento, a esta se le llama **Variable Latente**, o variable oculta.

Se puede estimar o inferir a que clase pertenece calculando:

$$z_i^* = \operatorname{argmax}_k p(z_i = k | \mathbf{x}_i, D),$$

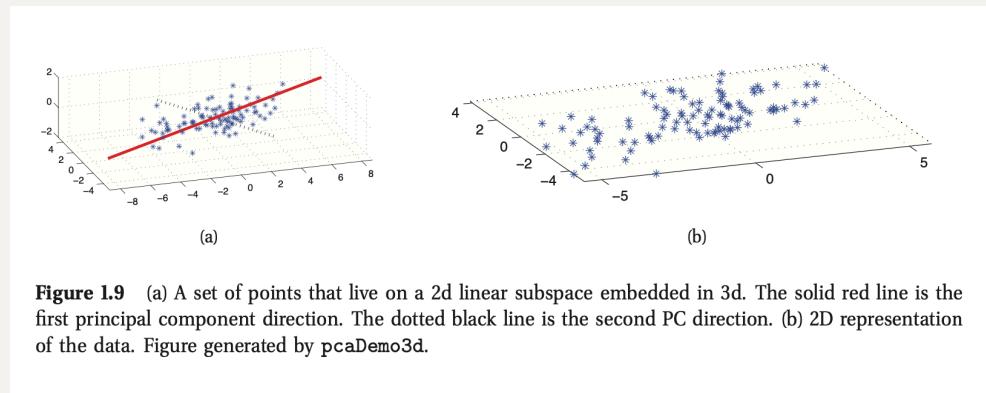
Que significa, calcular la clase k que que maximiza la función de probabilidad dado un dato \mathbf{x}_i en el conjunto de datos D

Ejemplo:



Descubriendo variables (o factores latentes)

- Útil cuando los datos son de dimensiones muy altas.
- Es útil para conocer el subconjunto de variables de un vector de características que conservan la máxima información. (Podemos reconstruir la información a partir de información clave)
- Se proyectan en un espacio de dimensión reducida $M : R^m \rightarrow R^n$
- Por lo regular $2 \leq n \leq 3$ para poder analizar los datos visualmente.
- Util para compresión de imágenes o aplicar algoritmos de agrupamiento.
- Uno de los métodos más utilizados es análisis de componentes principales o PCA.



Explicaciones conceptuales

Modelos paramétricos y no paramétricos

Modelos paramétricos.

- Son modelos que tienen un número fijo de parámetros independientemente de la cantidad de datos.
- Requiere un conocimiento sobre el fenómeno más detallado.
- Se apoya de suposiciones fuertes sobre los datos. (e.g. Datos distribuidos con cierta función de probabilidad)
- Funciona bien con pocos datos.
- No sufren de **sobre-ajuste**.

Ejemplos

1. Regresión lineal.
2. Estimación de distribuciones de probabilidad.

Modelos No-paramétricos:

- Los parámetros crecen con los datos.
- Se requiere un conocimiento menos profundo sobre el fenómeno.
- No requiere hacer suposiciones fuertes sobre los datos.
- Requiere muchos datos para tener buen desempeño
- Sufren de sobre-ajuste

Ejemplos:

Redes neuronales (Supervisado)

Redes neuronales convolucionales (Supervisado)

k -vecinos cercanos (Supervisado)

PCA (no-supervisado)

Mapas auto-organizados (No paramétricos - No supervisados)

Ademas de los modelos de clasificación mencionados existen otros modelos clasificadores que también son de importancia en este curso

- Minima distancia
- Modelo del Paralelepípedo
- Árboles de clasificación
- Clasificación por máxima verosimilitud

Ejemplos de Modelos paramétricos para clasificación.

Clasificador Lineal

Un clasificador lineal es una función de la forma

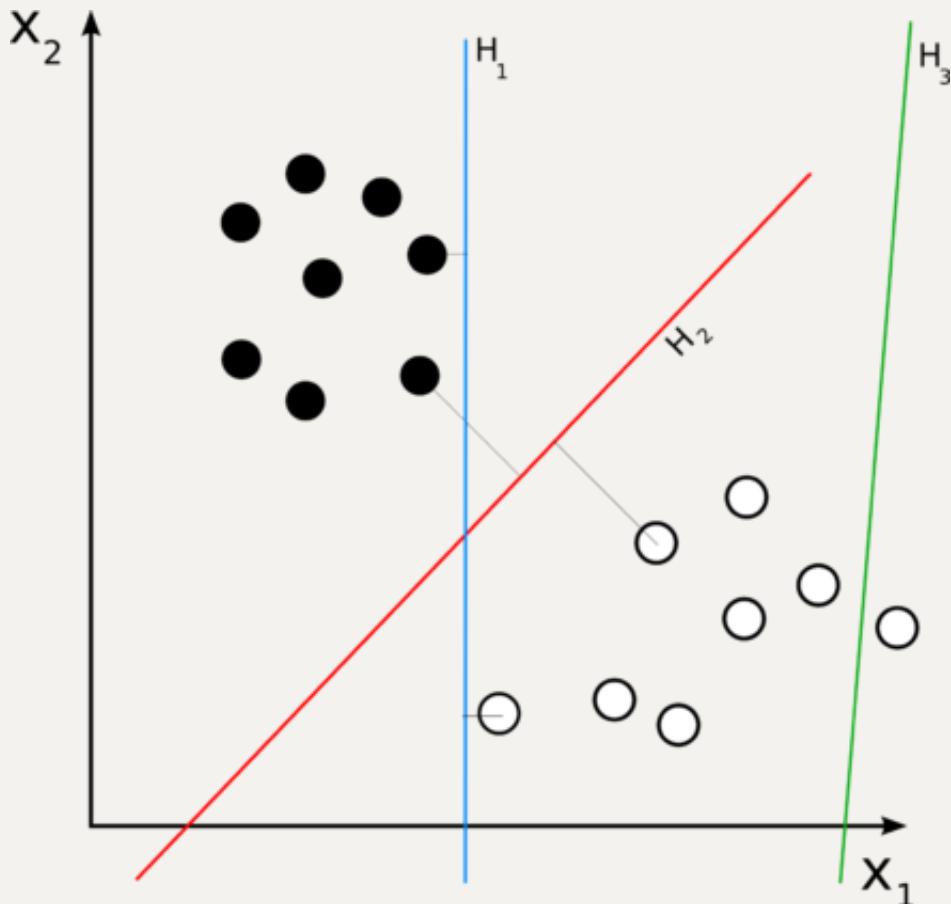
$$y = f(\mathbf{w}^T \mathbf{x} + b)$$

donde , \mathbf{w} es un vector de pesos, \mathbf{x} es el vector de entrada y la función f convierte el producto interno de los dos vectores en una salida.

la función f toma valor de:

- 1: si $\mathbf{w}^T \mathbf{x} + b > 0$, y
- 0: de otra manera.

Para el problema de clasificación de dos clases, el problema puede interpretarse como un hiperplano que divide el espacio de vectores de entrada x , en dos. Donde los valores que quedan de un lado del hiperplano son clasificados como positivos y lo que quedan debajo negativos



Regresión Logística

Es una generalización de la regresión lineal para clasificación binaria.

La respuesta o salida y de la función responde a una distribución de probabilidad de Bernoulli.

$$p(y|\mathbf{x}, \mathbf{w}) = Ber(y|\mu(x))$$

donde la respuesta toma dos valores: $y \in \{0, 1\}$. y $\mu(\mathbf{x}) = p(y = 1|x)$ es la probabilidad de éxito dada una entrada \mathbf{x} .

La función de distribución de probabilidad está dada por la función sigmoide:

$$\mu(\mathbf{x}) = \frac{1}{1+exp(\eta)}$$

Acoplando los términos en orden obtenemos la **regresión logística**

$$p(y|\mathbf{x}, \mathbf{w}) = Ber(y|sigm(\mathbf{w}^T \mathbf{x}))$$

La distribución de probabilidad de Bernoulli es discreta y dicotómica que calcula la probabilidad de éxito.

Ejemplo:

$$p(y_i = 1|x_i, \mathbf{w}) = sigm(w_0 + w_1 x_i)$$

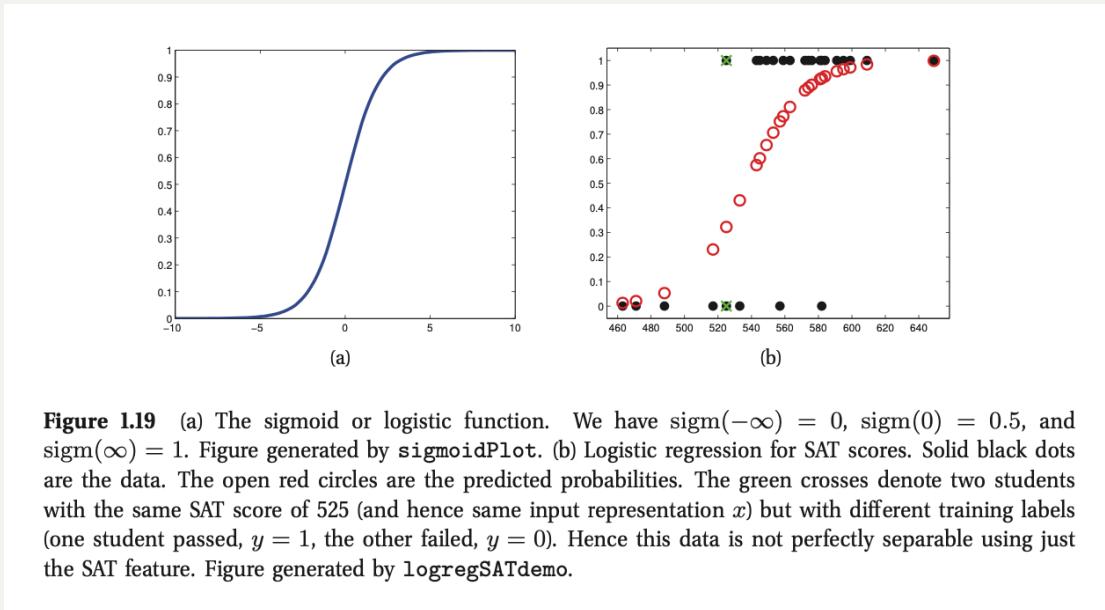
es aplicado a definir si un estudiante pasa la prueba ($y = 1$) o no ($y = 0$) de acuerdo a su puntuación.

Para esto a partir de los datos (x_i, y_i) , donde $x \in R$ es la calificación o puntaje representado por el eje horizontal, y $y \in 0, 1$ (pasa o aprueba la clase), representado por el eje vertical.

Los puntos negros son los datos reales.

Los rojos son salidas estimadas $p(y = 1|\mathbf{x}_i, \hat{\mathbf{w}})$

Si el umbral es mayor a 0.5 entonces la salida estimada será $\hat{y}(x) = p(y = 1|\mathbf{x}) > 0.5 = 1$



Ejemplo de modelo no paramétrico

k vecinos cercanos

Este algoritmo consiste en buscar aquellos datos cercanos a un *dato de prueba* dado un radio r o tamaño de vecindad K . Y regresa la proporción de clases encontradas.

Este algoritmo puede expresarse como

$$p(y = c | \mathbf{x}, D, K) = \frac{1}{K} \sum_{i \in N_k(\mathbf{x}, D)} I(y_i = c)$$

Donde $N_k(\mathbf{x}, D)$ son los índices pertenecientes al vecindario.

$I(e)$ es la función indicadora que devuelve 1 si e es verdadero y 0 en caso contrario.

con $K = 1$ se genera un diagrama de Voronoi.

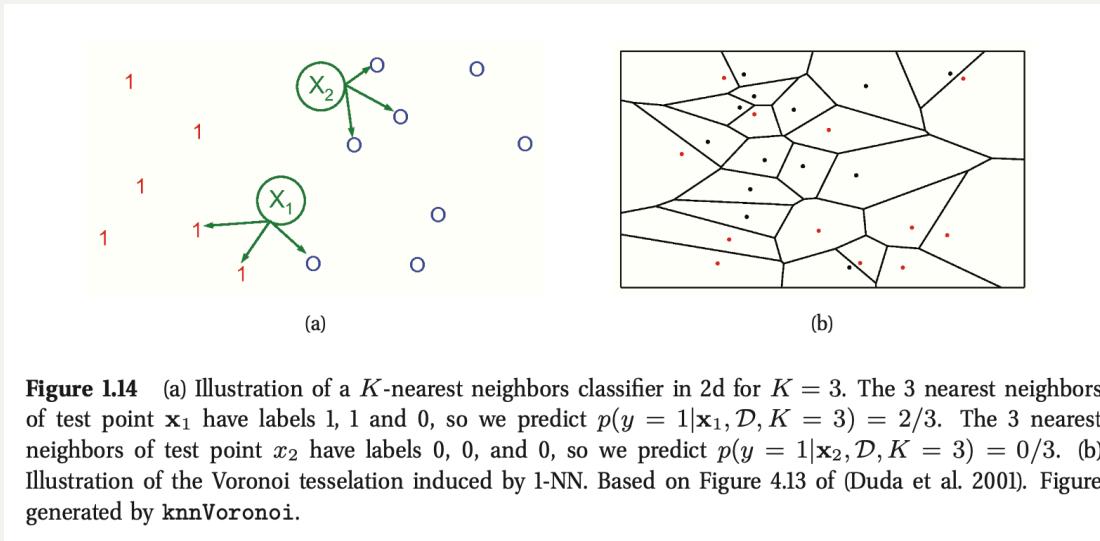


Figure 1.14 (a) Illustration of a K -nearest neighbors classifier in 2d for $K = 3$. The 3 nearest neighbors of test point x_1 have labels 1, 1 and 0, so we predict $p(y = 1 | \mathbf{x}_1, \mathcal{D}, K = 3) = 2/3$. The 3 nearest neighbors of test point x_2 have labels 0, 0, and 0, so we predict $p(y = 1 | \mathbf{x}_2, \mathcal{D}, K = 3) = 0/3$. (b) Illustration of the Voronoi tessellation induced by 1-NN. Based on Figure 4.13 of (Duda et al. 2001). Figure generated by knnVoronoi.

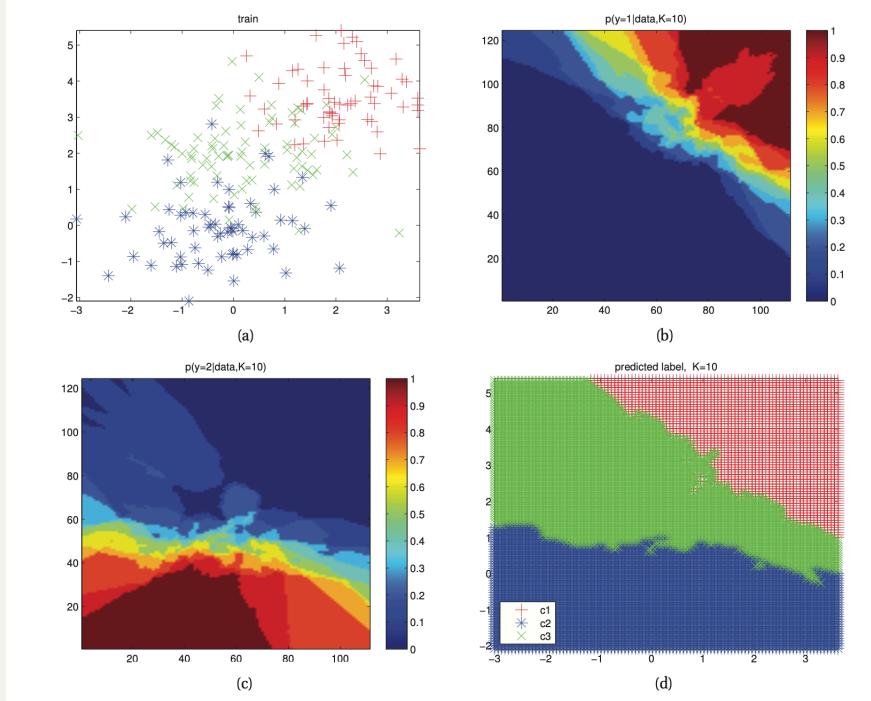


Figure 1.15 (a) Some synthetic 3-class training data in 2d. (b) Probability of class 1 for KNN with $K = 10$. (c) Probability of class 2. (d) MAP estimate of class label. Figure generated by `knnClassifyDemo`.

1) Datos sintéticos de 3 clases

2) Probabilidad de cada punto de pertenecer a clase 1 con $k=10$ vecinos cercanos.

3) Probabilidad de cada punto de pertenecer a clase 2 con $k=10$ vecinos cercanos.

4) Áreas de cada clase construidas con los mismos datos. Estas representan la clase con mas probabilidad de pertenecer a un dato \mathbf{x} . Se puede expresar como:

$$\hat{y} = \text{argmax}_c(y = c|\mathbf{x}, D)$$

Maldición de la dimensionalidad.

Se refiere a la incapacidad o el mal desempeño de los modelos de aprendizaje automático de poder escalar con los datos debido a la alta dimensionalidad del problema.

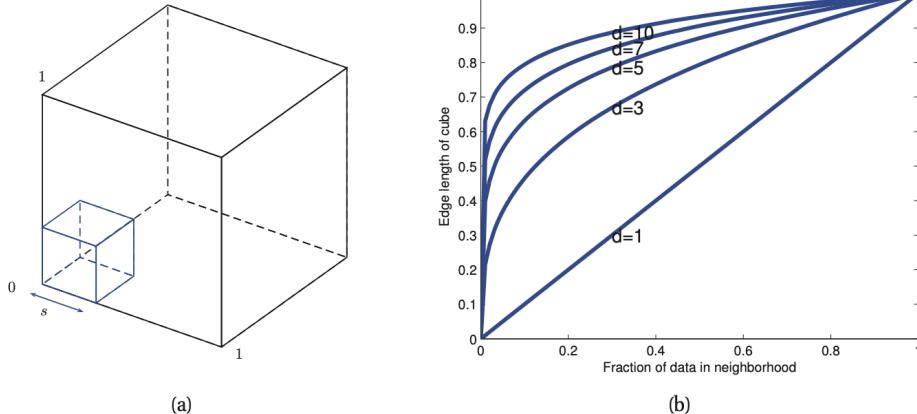


Figure 1.16 Illustration of the curse of dimensionality. (a) We embed a small cube of side s inside a larger unit cube. (b) We plot the edge length of a cube needed to cover a given volume of the unit cube as a function of the number of dimensions. Based on Figure 2.6 from (Hastie et al. 2009). Figure generated by `curseDimensionality`.

En el caso de vecinos cercanos, la distancia para obtener una vecindad mínima crece exponencialmente con el número de dimensiones.

En optimización el espacio de búsqueda crece exponencialmente con el número de dimensiones del espacio.

Etapas del proceso de clasificación (miércoles 3 marzo)

0. Definir el problema de clasificación (o regresión).
 - a. Clasificar Flores
 - b. Clasificación de Hongos (Comestibles o no comestibles)
 - c. Diagnosticar Cancer.
1. Colectar la mayor cantidad de datos posible con la mejor calidad. Ordenarlos en una tabla (como .csv)
2. Preparar los datos.
 - a. Aleatorizar los datos para que el orden entre ellos no afecte en el entrenamiento del modelo.
 - b. Hacer un análisis exploratorio de los datos para detectar características generales de las correlaciones, variables de interés, balance de clases, formas generales de las distribuciones.
 - c. Reducir la dimensionalidad de ser posible.
 - d. Definir el conjunto de entrenamiento y prueba.
3. Entrenar y probar varios modelos bajo las mismas condiciones.
4. **Seleccionar el modelo**
5. En caso de no tener resultados satisfactorios, ajustar parámetros o incluir nuevos modelos.

6. Predicción, clasificación o inferencia (Se puede hacer sobre un conjunto de datos de validación).
7. Interpretar el modelo.

Selección de modelo

Cuando tenemos una variedad de modelos, existe la necesidad de seleccionar el mejor.

Para esto se requiere calcular un **error generalizado**, el cual es el error esperado de la tasa de clasificaciones incorrectas de los datos futuros o no observados.

Para esto es necesario organizar los datos en 1 o mas particiones, donde cada particion estará compuesto de un conjunto de datos de entrenamiento, y el restante de prueba (Fig 1.21).

Una manera de validar el modelo es el siguiente:

0. Seleccionar el modelo de clasificación
1. Particionar los datos en al menos dos conjuntos para ajustar o entrenar un modelo:
 - a. Entrenamiento
 - b. Prueba
2. Para la partición: entrenar un modelo con el conjunto de entrenamiento y probar su ajuste con con el conjunto de Prueba.
3. Medir el error primero con el conjunto de entrenamiento y prueba. El error se puede expresar como:

$$err(f, D) = \frac{1}{N} \sum_{i=1}^N I(f(\mathbf{x}_i) \neq y_i)$$
4. (Opcional pero recomendado) Repetir el procedimiento varias veces seleccionando los conjuntos de manera diferente.

El error generalizado se expresa como el error promedio de un conjunto de datos de varias particiones diferentes dado por.

$$\text{error general} = \frac{1}{M} \sum_{i=1}^M err_i(f, D)$$

Donde M es el número de particiones diferentes.

En cada partición, el conjunto de entrenamiento es utilizado para ajustar parámetro del modelo, y el conjunto de prueba para calcular el error y evitar seleccionar un modelo **sobre ajustado**.

Este error puede ser aproximado utilizando un conjunto de datos de prueba.

Ejemplo de Validación Cruzada con un ejemplo de k vecinos cercanos.

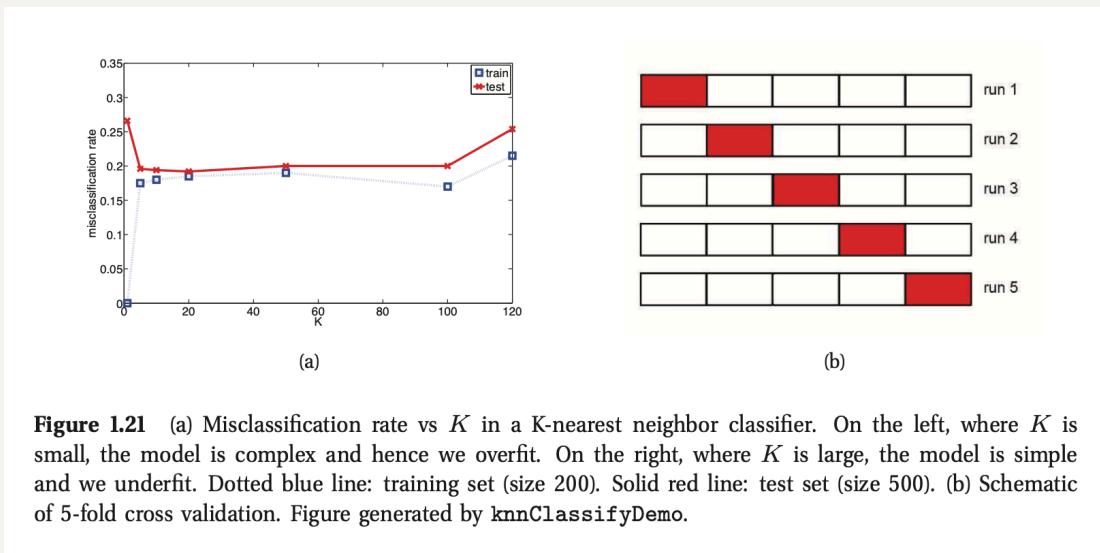


Figure 1.21 (a) Misclassification rate vs K in a K-nearest neighbor classifier. On the left, where K is small, the model is complex and hence we overfit. On the right, where K is large, the model is simple and we underfit. Dotted blue line: training set (size 200). Solid red line: test set (size 500). (b) Schematic of 5-fold cross validation. Figure generated by `knnClassifyDemo`.

Adicionalmente es posible reservar un 3er conjunto exclusivo llamado **conjunto de validación**. El cual es utilizado para corroborar la generalización del modelo. Los Datos de este conjunto de validación pueden mantenerse invariantes.

Sobre ajuste

Es un problema que sufren principalmente los modelos pertenecientes a aprendizaje automático. Un ajuste *perfecto* sobre el conjunto de entrenamiento impide desempeñarse de manera adecuada con cualquier otro dato que no haya sido contemplado en su entrenamiento. **Esto se detecta cuando tenemos un error de ajuste muy bajo en el conjunto de prueba y un error muy alto en el conjunto de entrenamiento.**

No Free Lunch Theorem

Se refiere a la no existencia de modelos Universales.

Todos los modelos estan mal pero algunos son útiles. George Box (Box and Draper 1987)

Aunque probemos nuestro modelo y lo validemos con los métodos adecuados para obtener cierta capacidad de generalización en cierto dominio de aplicación, este se desempeñara pobre en otros dominios diferentes.

Tarea 3 Experimento validación cruzada

Replicar el ejemplo de la figura 1.21 pero con diferente conjunto de datos. Estos pueden ser sintéticos, o descargados de alguna base de datos.

Dudas:

1)

Trabajar con vectores bidimensionales para poderlos graficar y observar sin dificultad en un plano 2-d.

Hacer validación cruzada. con al menos 2-iteraciones (2-fold validation).

Como producto generar una figura parecida a 1.15(d), 1.21 (a). e identificar con que parámetro existe sobre ajuste, y sub-ajuste.

Dudas por correo electrónico rdglpz@gmail.com o en clase

Entrega para el viernes 12 Marzo

Se pueden apoyar del siguiente código en Python:

<https://www.aprendemachinelearning.com/7-pasos-machine-learning-construir-maquina/>

https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py

[https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)#/media/File:K-fold_cross_validation_EN.svg](https://en.wikipedia.org/wiki/Cross-validation_(statistics)#/media/File:K-fold_cross_validation_EN.svg)

Unidad 2 Interpretación Geométrica de la Clasificación

Viernes 5 de Marzo

Notas: Correcciones y aclaraciones en la notación están escritas con un * con respecto a la última versión.

La regresión logística es un modelo estadístico que utiliza una **función sigmoidal** para modelar variables dependientes binarias.

En términos geométricos, regresión logística trata de encontrar un plano que separe las clases **minimizando el error de clasificación**.

Trabaja con conjunto de datos (dataset) casi separables.

Para clasificación binaria, si un hiperplano puede separar razonablemente bien dos clases, entonces se dice que el conjunto de datos es **linealmente separable**.

Derivación de la interpretación geométrica

Ejemplo:

Imaginemos que tenemos un conjunto de datos $D \in \{x_i, y_i\}_{i=1}^N$ donde $y_i \in \{0, 1\}$ y $x_i \in R^{Dim}$

Para este conjunto de datos necesitamos una función (lineal) que nos ayude a separar los datos en dos clases.

Para esto recurriremos a la ecuación general de la recta que separa dos clases:

$$ax + by + c = 0$$

donde en aprendizaje automático para un hiperplano cualquiera se escribe como:

$$\mathbf{w}^T \mathbf{x}_i + w_0 = 0,$$

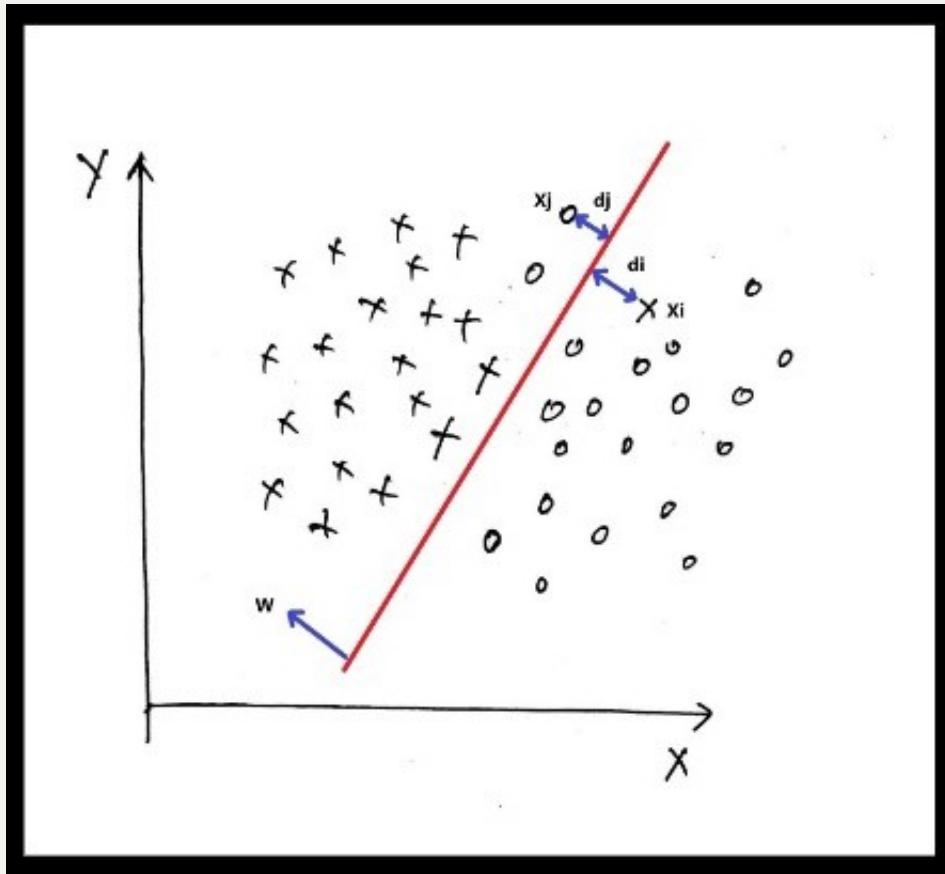
y queremos encontrar los mejores parámetros \mathbf{w} . que nos ayuden a **maximizar la precisión** (minimizando el error) de la clasificación en el conjunto de datos D

Si la línea pasa sobre el origen del hiperplano entonces el intercepto $w_0 = 0$

Problema:

Dado un conjunto $D \in \{x_i, y_i\}_{i=1}^N$ donde la respuesta esta dada por una variable binaria $y_i \in \{-1, 1\}$ (*) y la variable independiente es $x_i \in R^{dim}$, encontrar un hiperplano que pueda separar ambas clases de tal manera que también pueda clasificar correctamente nuevos datos.

Ejemplo "hiperplano" bidimensional



Los pesos \mathbf{w} se representan geométricamente en un plano como un punto $P = (w_0, w_1)$, el cual a su vez es la terminación un vector con dirección ortogonal a la frontera de decisión.

Ejemplo *: Si la frontera de decisión esta dada por la recta.

$$m_f w_0 = w_1$$

$$(m_f w_0 - w_1 = 0)$$

donde $m_f = 2$ (pendiente de la frontera) entonces:

$$2w_0 = w_1.$$

La pendiente ortogonal correspondiente esta dado por

$$m_o w_0 = w_1$$

$$\text{donde } m_o = -1/2$$

$$-1/2w_0 = w_1$$

Porque en el plano bidimensional se cumple que

$$m_f m_o = -1$$

$$\frac{2}{1} \cdot -\frac{1}{2} = -1$$

Ejercicio: Si tenemos pesos $w_1 = 2$, y $w_2 = 3$. Cual sería la pendiente de la frontera de desición?

Para probar a que clase pertenece un punto x_i es necesario acordarnos de la fórmula para medir la distancia mas cercana de un punto x_i a la recta.

(Distancia Sin intercepto)

$$d_i = \frac{\mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|}$$

(Distancia con intercepto)

$$d_i = \frac{\mathbf{w}^T \mathbf{x}_i + w_0}{\|\mathbf{w}\|}$$

donde $\|\mathbf{w}\| = (\sum_{i=1}^{Dim} w_i^2)^{1/2}$ **es la magnitud o módulo del vector** . * (Aclaración en la definición)

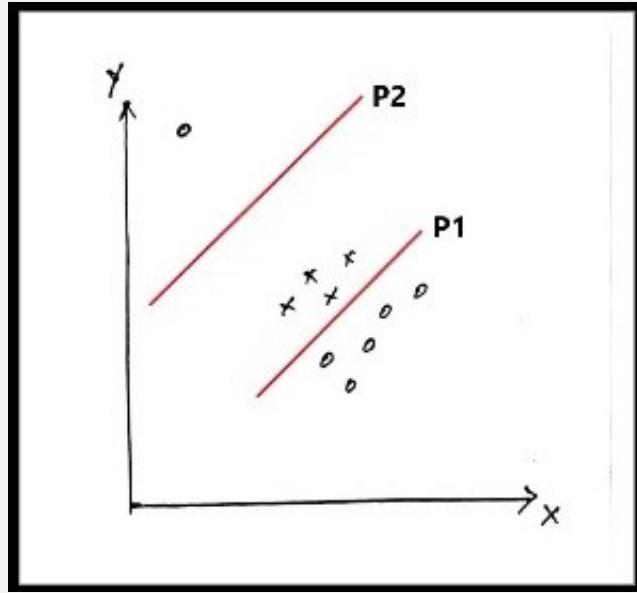
Por lo tanto si $d_i > 0$ entonces x_i pertenece a la clase positiva.

Una primera formulación intuitiva de la función objetivo es la siguiente:

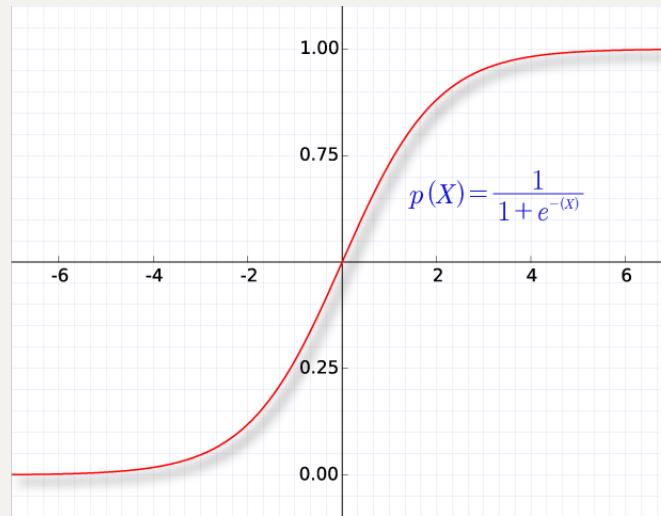
$$\hat{\mathbf{w}} = argmax_{\mathbf{w}} \sum_{i=1}^N (y_i \cdot (\mathbf{w}^T \mathbf{x}_i + w_0))$$

Esta función estará sumando valores positivos siempre y cuando se cumpla
 $sign(\mathbf{w}^T \mathbf{x}_i + w_0) = sign(y_i)$

Problema de esta formulación: Es sensible a los datos atípicos (outliers).



Solución: Usar una función que mapea $f : y_i \cdot (\mathbf{w}^T \mathbf{x}_i + w_0) \rightarrow \{0, 1\}^*$



$$\text{sigm}(z) = \frac{1}{1 + \exp(-z)}$$

Ejemplo funciones sigmoidales con dos parámetros.

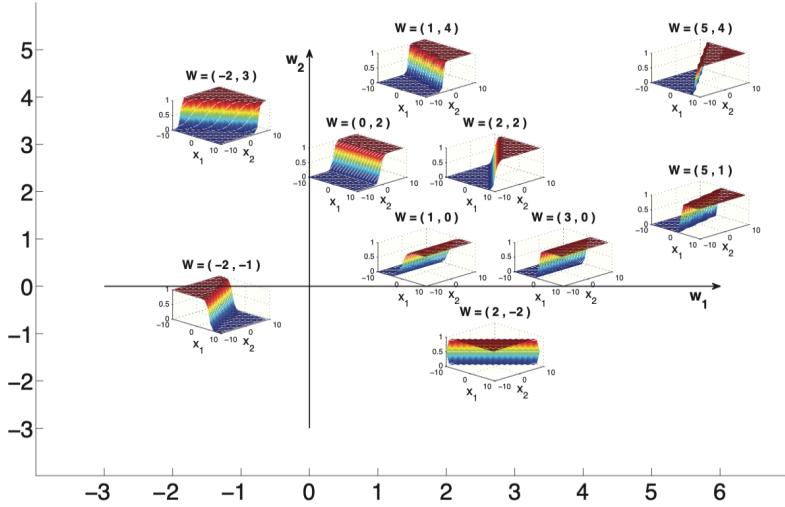


Figure 8.1 Plots of $\text{sigm}(w_1x_1 + w_2x_2)$. Here $\mathbf{w} = (w_1, w_2)$ defines the normal to the decision boundary. Points to the right of this have $\text{sigm}(\mathbf{w}^T \mathbf{x}) > 0.5$, and points to the left have $\text{sigm}(\mathbf{w}^T \mathbf{x}) < 0.5$. Based on Figure 39.3 of (MacKay 2003). Figure generated by `sigmoidplot2D`.

Esta función regresa valores entre 0.5 y 1 cuando $y_i(\mathbf{w}^T \mathbf{x}_i + w_0) > 0^*$

Esta función regresa valores entre (0, 0.5) en caso contrario.

Por lo tanto la nueva formulación se expresa como

$$\hat{\mathbf{w}}, \hat{w}_0 = \text{argmax}_{\mathbf{w}, w_0} (\sum \text{sigm}(y_i \cdot (\mathbf{w}^T \mathbf{x}_i + w_0)))$$

Problema: Tiene poca estabilidad numérica debido a los parámetros (variables) \mathbf{w} en el exponente en el denominador.

Solución: Aplicar logaritmo.

$$\hat{\mathbf{w}}, \hat{w}_0 = \text{argmax}_{\mathbf{w}, w_0} (\sum \log(\frac{1}{1+\exp(-y_i \cdot (\mathbf{w}^T \mathbf{x}_i + w_0))}))$$

(paso intermedio, completar por ustedes mismos)

$$\hat{\mathbf{w}}, \hat{w}_0 = \text{argmax}_{\mathbf{w}, w_0} (\sum -\log(1 + \exp(-y_i \cdot (\mathbf{w}^T \mathbf{x}_i + w_0))))$$

Último paso, convertirlo en un problema de minimización.

(Completar ustedes mismos)

Referencias

Machine Learning: a Probabilistic Perspective. by Kevin Patrick Murphy. MIT Press, 2012.

<https://towardsdatascience.com/geometric-interpretation-of-logistic-regression-4f85047a5860>

https://mathinsight.org/distance_point_plane_examples

<https://quimicayalgomas.com/wp-content/uploads/2015/03/logaritmos-propiedades.png>

Unidad 3 Clasificación No supervisada

Miércoles 10 de Marzo

Definición y formulación matemática

Examen Unidad 1,2 y 3

Viernes 12 de Marzo

Bibliografía

https://www.um.es/geograf/sigmur/temariohtml/node74_mn.html

