

Inteligencia Antiespecial

Tema -7-

- El aprendizaje no es una característica humana, engloba a todo
- El aprendizaje automático estudia como extraer conocimientos de los datos

• Def. Machine learning (Aprendizaje automático)

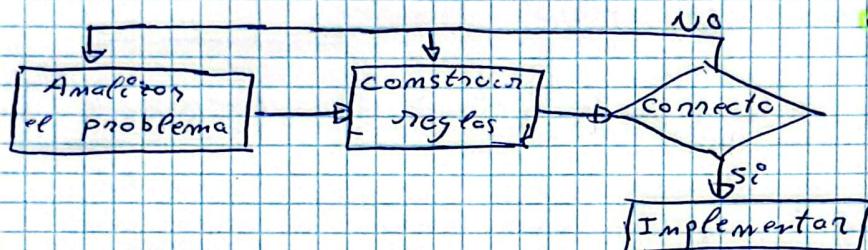
Un programa se dice que aprende de la **experiencia (E)** respecto a una clase de tareas (**T**) y con un **comportamiento (P)** si el comportamiento en las tareas **T**, medida por **P** mejora con la **experiencia E**.

T Task

- ML simplifica la toma de decisiones

P Performance

E Experience



- ML algorithm has 3 basic components:

- Representación (tipo de modelo que queremos)
- Evaluación (goal)
- Optimización (del propio algoritmo)

- Para conseguir la optimización, existen los siguientes modelos de algoritmos

• Deterministas

• Heurística

Algoritmos predictivos: Queremos saber lo que va a pasar en el futuro

Algoritmos descriptivos: Entender la relación oculta en el dato

- Ninguno es mejor que el otro, y dan lugar a distintos tipos:

• Algoritmos Supervisados

• " " No supervisados

• " " Semi supervisados

La estadística y ML son similares

- Estadística descriptiva: descripción de los datos para ser comprensibles

- Inferencia estadística: técnicas que permitan generalizar algunas muestras de datos

Deep learning / Estadística

Pesos sinápticos = Parámetros

Aprendizaje = Estimación

Red neuronal = Modelo

Aprendizaje supervisado = Regresión

" " No " = Clasificación

Performance = Función de pérdida

Típos de aprendizaje

- **Supervisado**: Tratamos de encontrar la relación entre una variable dependiente (y), y una variable explicativa (x); Usando un conjunto de datos etiquetado, ó independiente
 - para cada ejemplo, tenemos que ir diciendo a qué corresponde cada etiqueta.

EJ: → Reconocimiento de caracteres

En cualquier modelo

- Existen 2 fases:

- fase de aprendizaje: Mostramos un conjunto de elementos al modelo para que aprenda
- fase de predicción: Presentamos un ejemplo que no ha visto para intentar que nos de un resultado correcto
- Las variables independientes o predictivas permiten determinar las respuestas ó variables dependientes
- En la fase de entrenamiento tendremos unas observaciones (x, y) con unos parámetros Θ , buscamos un modelo f

$$y = f_{\Theta}(x)$$

- A la hora de predecir, cambia ligeramente

$$\hat{y} = f_{\Theta}(x')$$

x' = ejemplos des conocidos

Θ = vamos a trabajar sobre los parámetros óptimos ya encontrados

\hat{y} = no son datos reales, sino predicciones

- Con todo esto podemos saber nuestra performance del modelo.

$$\text{Performance} = g(\hat{y}, y)$$

nuestra mejora

- Un problema común es la reducción de la dimensionalidad para facilitar el aprendizaje del modelo.

- El problema de transformar datos en crudo en algo útil se llama ingeniería de características

Tema - 1-
Inteligencia artificial:

- "Habilidad de tener conocimiento en algo creado por los humanos"

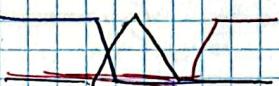
$$IA \neq ML = DL$$

{ Deep Learning (DL): Aprendizaje basado en redes neuronales profundas
ML: Habilidad de aprender sin estar explícitamente programado
IA: Ingeniería de hacer máquinas y programas inteligentes

IA

• Primitiva: Máquinas de inferencia $P \rightarrow q \Leftrightarrow T \rightarrow P$

• Intermedia: Lógica difusa, permite tomar decisiones



A: 0'5 descriptable Lógica
B: 0'5 de " " probabilidad difusa

• Moderna: Deep learning y Algoritmos genéticos

Tema - 3-

• Aprendizaje no supervisado:

No sabemos las etiquetas de los ejemplos x y el objetivo es clasificarlos en K categorías donde K puede ser un número especificado o no.

- El sistema intenta aprender sin un "profesor"

ejm → clasificación 3 tipos de flores

ejm → chatbot, Tenemos que detectar la intención del cliente y agruparlo con los que tengan el mismo interés

- Una aplicación particular puede ser los algoritmos de visualización, para observar los datos de una forma gráfica.

- Suelen utilizarse como un paso previo a la reducción de dimensionalidad

- Otra aplicación puede ser la detección de anomalías y se eliminan para no distorsionar el proceso normal de construcción del modelo.

• Aprendizaje Semi-supervisado:

- Se utiliza cuando tenemos un dataset etiquetado y no etiquetado, y el número de lo no etiquetado es mayor que lo que se está
- Consiste en 2 pasos:

- 1.- Se agrupan los ejemplos según las características compartidas.
- 2.- Despues de que uno de los ejemplos del grupo es etiquetado, el resto de los ejemplos es también etiquetado

• Aprendizaje por refuerzo:

El objetivo consiste en utilizando observaciones obtenidas al interactuar con el entorno y tomar acciones que maximicen alguna recompensa.

- El sistema de aprendizaje es llamado agente. El agente observa el entorno y realiza algunas acciones.
- El agente obtiene recompensas o penalizaciones
- Debe aprender solo para obtener recompensas durante el mayor tiempo. La mejor estrategia,
- La diferencia de esta en comparación con los otros 2 modelos es que las recompensas se retrasan.
- Un comportamiento pasado afecta al futuro

→ Para todos los tipos de aprendizaje, el aprendizaje puede ser incremental o secuencial.

- El primero aprende de forma online mientras que el segundo lo referimos a aprendizaje por lotes o offline

• El aprendizaje online es incremental y recibe pequeños grupos llamados mini-lotes.

Tema - 4 -

• Modelos Lineales

- Para ligar 2 variables, (x, y) , en la que y depende de x .

$$y = \alpha + \beta x \quad \begin{cases} \text{? esto es un modelo de relación } x \text{ con } y \\ \text{parámetros} \end{cases}$$

- El proceso de encontrar los parámetros óptimos para el modelo se llama:
estimación o aprendizaje

- Algoritmos son procedimientos computacionales (Normalmente iterativos)

- Los modelos pueden tener algún componente ϵ (error) de modo que haya una respuesta incluso si la entrada es cero:

$$y = \alpha + \beta x \quad \text{* Si no tomo café, puedo estudiar 40 horas}$$

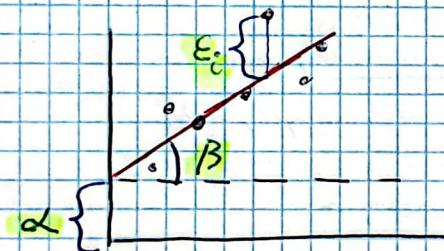
• Y podemos pronosticar nuevas observaciones:

$$\hat{y} = \hat{\alpha} + \hat{\beta} x' \quad \begin{cases} \text{(llamado modelo lineal)} \end{cases}$$

- Podemos decir que existe una relación lineal entre y, x

- Específicamente, es un modelo univariado ya que solo tiene una variable independiente

- β es la pendiente



• Debemos tener en cuenta también el error (ϵ)

$$y = \alpha + \beta x + \epsilon$$

• Error medio cuadrático

$$\epsilon = g(y, \hat{y}) = (y - \hat{y}) = (y - (\alpha + \beta x))$$

- función de pérdida

• Tengo que buscar α, β para obtener el error mínimo

$$\min_{\alpha, \beta} \sum \epsilon_i^2 = (y - (\alpha + \beta x))^2$$

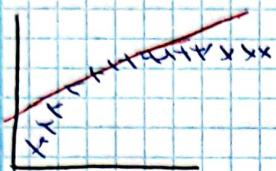
(2) → para evitar errores

• Ecuación normal en caso lineal

$$\alpha, \beta = (X^t X)^{-1} X^t Y$$

Modelo no lineal

- En ocasiones el modelo lineal no se adapta bien y es más conveniente usar un modelo de regresión cuadrática



$$Y = \alpha + \beta t + \gamma t^2$$

- Modelo Logístico multi variable

$$Y = \frac{1}{1 + e^{-B_0 - B_1 x_1 - B_2 x_2 \dots}} \quad (\text{si } Y=1)$$

$Y=1$ si ocurre el evento

Modelos no paramétricos y paramétricos

Existe una infinidad de formas distintas de calcular un modelo

- Paramétricos

• Nº fijado de parámetros

- No paramétricos

• Nº indeterminado de parámetros

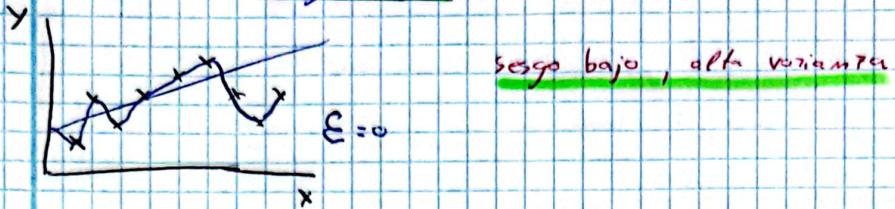
- Gracias a la propiedad de aproximación universal, podemos aproximar cualquier función a cualquier nivel de complejidad.

• por ejemplo el modelo polinomial de orden m

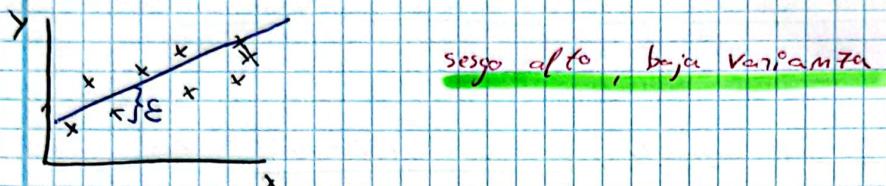
$$Y = \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3^2 + \alpha_4 x_4^2 + \dots$$

Tema -5- Sesgo y Varianza

- Si el modelo es no-lineal y representa exactamente los datos, se dice que tiene un sceso bajo.

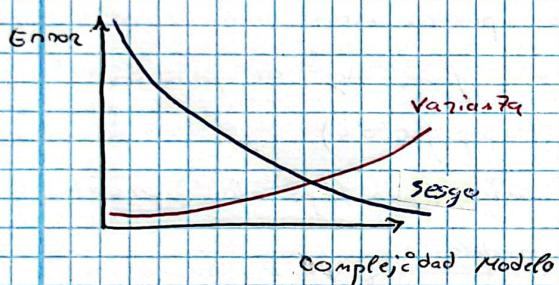


- Pasa lo contrario con un modelo lineal, no es capaz de capturar los datos de entrenamiento, sesgo alto



- cuando el modelo tiene una gran diferencia entre los datos y las predicciones se dice que tiene una alta varianza

-- A esto se le llama dilema sesgo-varianza --



- tipos de error

- Errores de modelo \rightarrow sesgo
- Errores imebitables \rightarrow Varianza

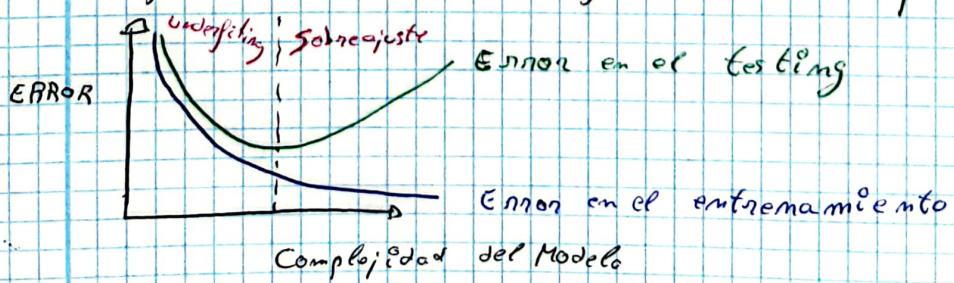
Sobreajuste y underfitting

- si el modelo es muy flexible, trabajará como una "base de datos" y únicamente recuerda el entrenamiento.

* Se produce un sobreajuste

- por el contrario, si el modelo es muy simple se puede producir un underfitting

* En general tendremos la siguiente curva de aprendizaje



• El balance óptimo entre seso y varianza es TEÓRICO

Medidas de pérdida y rendimiento

- La elección de la función de pérdida es crucial
 - Una mala función de pérdida \Rightarrow modelo malo aun teniendo buenos datos y suficientes
- generalmente podemos definir un problema de aprendizaje como encontrar los parámetros óptimos para minimizar diferencia entre la predicción y la real.
 - Una vez construido el modelo, podemos utilizar las funciones de performance
 - Nos permite evaluar los modelos posteriormente
 - Esto se hace por distintas razones:
 - { ① Las funciones de pérdida > Performance pueden estar relacionadas
 - ② Existen algoritmos muy eficientes para funciones de pérdida particulares (Algoritmo de retropropagación)
 - ③ La funciones de performance pueden ser algún tipo de pruebas estadísticas para comparar distintos modelos
 - ④ Usar funciones de performance en vez de pérdida puede evitar algunos errores de sobreajuste
 - Como dijimos, en los modelos lineales podemos utilizar el 'error cuadrático medio' (mse)

$$mse = g(y, \hat{y}) = (y - \hat{y})^2$$

- queremos calcular el mse para todas las observaciones $y_1, y_2, y_3, \dots, y_m$

$$mse = g(y, \hat{y}) = \frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}_i)^2 =$$

$$= \frac{1}{m} \sum_{i=1}^m (y_i - \beta_0 - \beta_1 x_i^1 - \beta_2 x_i^2 - \dots - \beta_m x_i^m)^2$$

Para minimizar el error

• msc tiene una serie de características que se deben tener en cuenta:

{ - msc es una medida si algún pronóstico es muy malo → msc es grande

- msc está expresado en unidades cuadradas, no en las mismas unidades que las variables que queremos predecir

④ Si lo que es una var, pero no una var al cuadrado ⚡

- podemos emplear la raíz del error cuadrático medio

$$\text{R.M.S.E.} = g(Y, \hat{Y}) = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}$$

→ el error medio absoluto (mae) (permite a lo largo del rango Y)

$$\text{m.a.e.} = g(Y, \hat{Y}) = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i|$$

- Para expresar el error en términos porcentuales, error absoluto porcentual medio

$$\text{m.a.p.e.} = g(Y, \hat{Y}) = \sum_{i=1}^n \frac{|Y_i - \hat{Y}_i|}{Y_i}$$

• En el caso de la claseficación binaria, msc puede no ser óptima

↑

como alternativa se ha propuesto la entropía cruzada

• En cuanto a las funciones de desempeño, puede usar la estadística R^2 [0, 1]

• Para la claseficación binaria, una de las medidas de performance más empleadas es! Matriz de Confusión

- tengo 2 clases $\{+, -\}$, $\{\text{bien}, \text{mal}\}$, $\{a, b\}$

		Recall	
		Positive	Negative
Predicción	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

TP: Predicimos que va a ser positivo y es efectivamente positivo.

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Exactitud} = \frac{100 + 70}{170 + 60} = 0'78$$

$$\text{Precisión} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

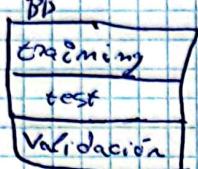
- Podemos utilizar F1 score para comparar 2 o más clasificadores

$$F1 = \frac{2 \cdot \text{recall} \cdot \text{Precision}}{\text{recall} + \text{Precision}}$$

- División de datos

Boosting

- Utilizar los datos para aprender y testear (incluso validar)

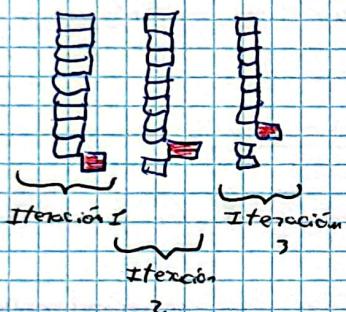


¿Cómo lo dividimos? 90/5/5?
80/10/10?

• Para evitar el problema de selección adversa y para obtener una estimación más fiable del error de validación cruzado basada en la idea de bootstraping.

- Dividir los datos en K columnas (popular choice $K=10$)

- $K-1$ para entrenar y 1 para testear



Regularización

• Dotar al modelo de conciencia

• Si no es capaz de generalizar \Rightarrow No es buen modelo

queremos un modelo:

- Que tenga capacidad de aprender \rightarrow ajuste

- " " " " permitir su capacidad \rightarrow complejidad

- Métodos de regularización Ridge y Lasso

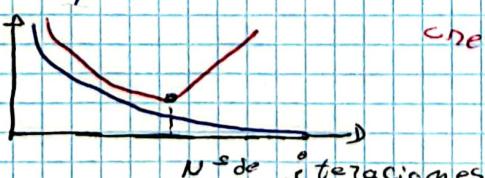
• Ridge: Es regresión con más parámetros pero que su valor es bajo ($0.01, 0.02, 0.07$)

• Lasso: Al contrario

$(0.7, 0)$

Cuando vea que el error empieza a crecer, parar

Nº de parámetros



Tema -6-

Importancia del preprocesamiento de datos y la ingeniería de características

- Antes de implantar un modelo predictivo, se ha de seguir el modelo ETL

{

- Extracción de la información de fuentes fiables
- Transformación de la info para ser usable con el modelo (en R&GB)
- Carga de la información en los sistemas

- Suelte ocupar este proceso el 80% del tiempo de desarrollo

- La mayor parte de modelos tienen un enfoque numérico
(se debe transformar la info en números)

• La calidad de los datos es esencial en ML y la calidad de los modelos dependerán de ellos.

• Las características de un determinado ejemplo = Features (características)

- Intentaremos trabajar con Features independientes y con poder discriminatorio, (que aporten algo).

• Ingeniería de características: Área del ML que intenta transformar los datos puros en Features que representen mejor el problema.

Intuición de datos

• Para desarrollar una visión intuitiva de los datos, se suele lograr gracias a la compresión de la info contenida en los datos

- Esto se puede conseguir si mostramos los datos (gráficas)

- ó gracias a los estadísticos, media, mediana, moda, ...

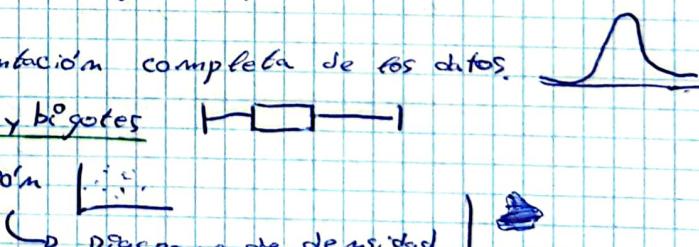
- puede ser útil saber si existe correlación, R^2 , entre 2 variables

• Los estadísticos pueden llegar a mostrar info. sesgada o equivocada

• Los histogramas nos dan una representación completa de los datos.

- otro podría ser el diagrama de caja y bigotes

- " " " Diagramas de dispersión



Pre-procesamiento de datos e Ingeniería de características

La utilización de estadísticos o representaciones puede contener algunos problemas.

- Falta de datos: Cuando algunos Features no están disponibles

- Puede existir algún dato que descomoscarmos
- " " " " " no sea aplicable
(Nº de piso, si vivo en chalet)

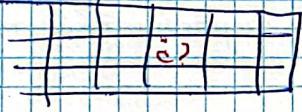
- Los modelos se verán afectados por esta falta de datos o no.

3 tipos:

- cuando los datos están perdidos ^{completamente} aletoriamente (MCAR)
- datos perdidos aletoriamente (MAR)
- Perdida NO aletoria (MNAR)

(o el valor perdido depende del valor de la variable)

Soluciones posibles:

- Borrado de filas: eliminar el ejemplo que le falte algún dato
- Eliminar la característica que contenga datos faltantes
- Imputación de datos: Intentar asignar un valor en donde falte
 - ... 
 - ¿Media?
 - ¿Moda?
 - ¿Seguir un patrón parecido?
- Utilizar el anterior
- El anterior más el siguiente entre 2

Transformación de datos: Proceso de modificar los datos para hacerlos más manejables para los algoritmos (también se llama Limpieza de datos)

- Una muy común es la Estandarización [$N(0, 1)$]

$$x' = \frac{x - \mu}{\sigma} \approx N(0, 1)$$

- podemos utilizarlo para uniformizar las escenas de un conjunto de datos
- otra opción puede ser transformar las variables entre [0, 1]

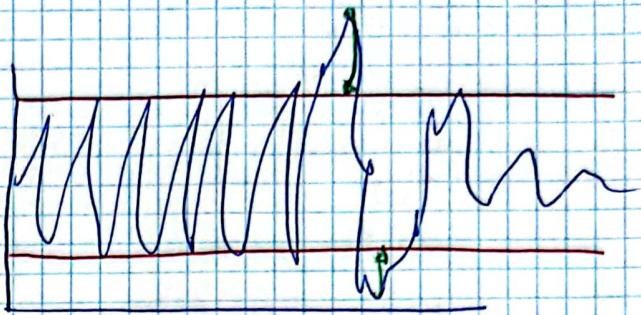
$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad [0, 1]$$

Outlier: Observación que se desvía tanto de las otras que levanta sospechas de que ha sido generada con un mecanismo estadístico diferente.

$$x = \begin{cases} \min(x, \mu + 2\sigma) \\ \max(x, \mu - 2\sigma) \end{cases}$$

Lo marca las 2 líneas rojas

- Cuando un dato supera una banda, consideramos que su valor es la outlier



Z-score

Una vez calculado, podemos eliminar los datos que tengan un resultado excesivo

$$Z\text{-score} = \frac{x - \mu}{\sigma}$$

■ Representación de los datos:

- La representación de unos datos puede llegar no ser tan adecuada.
- Una de las representaciones más usadas es. One Hot encoding
 - Creamos una variable "dummy" que será 1 en una determinada posición y el resto de posiciones tomará 0.

4 clases A,B,C,D

A	1	0	0	0
B	0	1	0	0
C	0	0	1	0
D	0	0	0	1

■ Tratamiento de datos incompletos

cuando no tenemos una observación acerca de un ejemplo

- Simulación de datos: A partir del resto de datos, intentaremos generar datos

$$O_t = O_{t-1} + \text{Random} \quad (\text{Muy empleado en la actualidad})$$

■ Datos desbalanceados

Cuando tenemos un dato que no está representado suficientemente en la base de datos

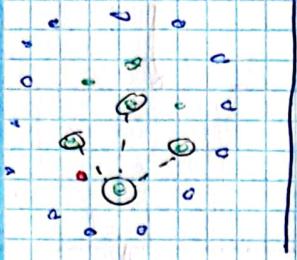
en 2BD 1º se utilitza el 95%

2º " " " 5%

Este problema existe en casi todos los problemas de ML

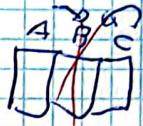
- Para solucionarlo, podemos intentar muestrear más fog que son suficientes.
- Se puede intentar crear datos sintéticos

- A la hora de crear datos sintéticos, existe un algoritmo para ello llamado SMOTE: (Synthetic Minority Oversampling Technique)



1 Reducción de los datos

- Se intenta reducir la carga computacional
 - por ejemplo eliminando resultados similares
 - " " " Features resultantes de otros



Regla de Ocam's

- Ante hipótesis igualmente razonables, se ha de elegir con menor N° de presunciones
- Dentro de la selección de características tenemos varias alternativas

- Métodos de filtrado: Eliminar los atributos sobrantes
 - Método de empacamiento: Se considera la selección de variables como problema de búsqueda.
 - Método de regularización: Penalizan modelos que emplean más características.

- La selección de características puede ser:

- Incremental
 - Decremental

Tema 8

Aprendizaje K-mm

y/n

- Recomendaciones de metrix

[gangsters] [el pacino] ... [Action] 

- Analogía: Relación de semejanza entre cosas distintas

Vecino próximo

• nosotros tenemos 2 patrones, A, B con una serie de características

- Para producir un nuevo patrón, miro el que tenga + características iguales

- A misma igualdad de características, puedo hacer la media de las 'Y' para sacar el resultado

					Y
A	1	4	3	9	
B	4	4	0	5	
C	1	4	0	?	$\frac{9+5}{2} = 7$

- Si no encaja ningún patrón, calculo la distancia y selecciono la menor

- construye un sistema no paramétrico

- también se llama aprendizaje vago

• Diccionario del algoritmo

① tiempo que categorizar los items

② calcular la distancia entre los 2 patrones (distancia euclídea) y sumarlos

③ calcular la raíz cuadrada

con esto soy capaz de saber la distancia entre 2 patrones

$$\underline{K-mm} = K \text{ vecinos próximos}$$

• Problemas

④ Dimensión maldad

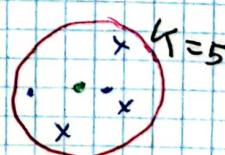
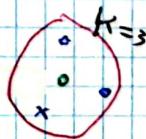
- Si es de alta dimensión, puede llegar a no tener vecinos (máquinas, medicina)

↳ se le llama "la maldición de la dimensionalidad"

⑤ Calibración

Una regla eurística popular es $K = \sqrt{n}$

$$\text{Círculo } K=2$$



③ Escalas de medida

- No es lo mismo usar ' m' o ' ft' '
- Cuando corre un algoritmo en otro país, puede dar resultados diferentes por la escala
- Para resolverlo: escalamos todo

④ La dependencia de las características

- Puede que una característica tenga correlación con otra y el modelo esté contando las 2
- Peso y altura

⑤ Importancia de las características

- No todas las características tienen el mismo peso
- Solución: Ponderar las características

⑥ Sensibilidad frente a características influentes

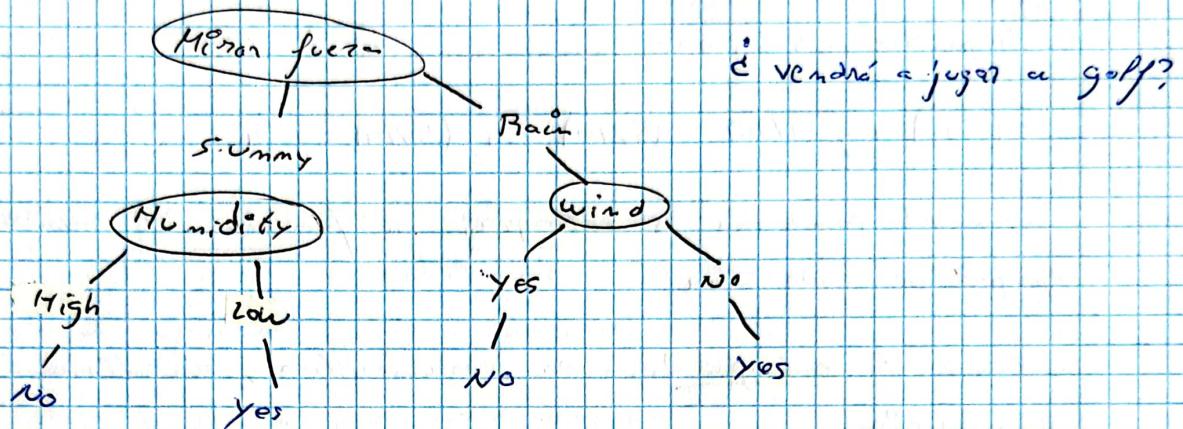
⑦ Complejidad computacional

- Altísimo coste de memoria

Tema 9

Arboles de regresión

- Son de tipo inducción
- Consumen muy poca memoria y son muy rápidos
- Son interpretables, puedes entender porque llegan a esa decisión
- Los ejemplos que deben clasificarse se llaman instancias
- Tendremos modos que tomarán decisiones binarias (Sí o No)
- Nosotros venimos el modo raíz y los modos terminales, lo del medio no

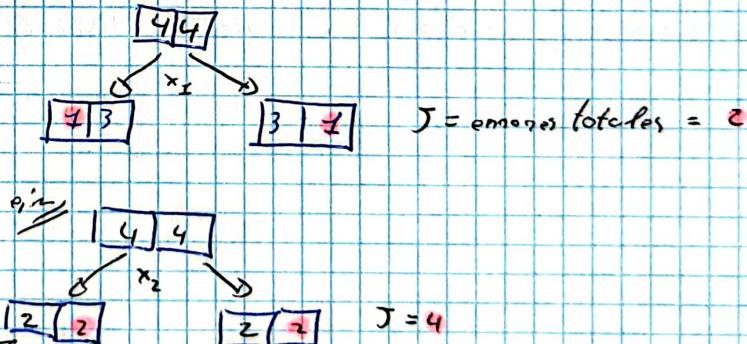


a los arboles se pueden transformar en reglas de inferencia

Aprendiendo el algoritmo

- No podemos hacerlo por fuerza bruta, necesitaremos un algoritmo
- Puedo el siguiente problema

x_1	x_2	x_3	y	ose tomo $x_2 = 0$, tendré 1 error, 3 aciertos		
0	0	0	1	• Si tomo $x_2 = 1$, " 3 errores, 1 acierto		
0	0	1	0			
0	1	0	1			
0	1	1	1			
1	0	0	0	<table border="1" style="display: inline-table;"><tr><td>4</td><td>4</td></tr></table>	4	4
4	4					
1	0	1	1	<table border="1" style="display: inline-table;"><tr><td>4</td><td>3</td></tr></table>	4	3
4	3					
1	1	0	0	<table border="1" style="display: inline-table;"><tr><td>3</td><td>4</td></tr></table>	3	4
3	4					
1	1	1	0	<table border="1" style="display: inline-table;"><tr><td>4</td><td>4</td></tr></table>	4	4
4	4					



en caso de que los 2 cuadros tengan el mismo valor, elijo el que

quiero

ojo

- A corto plazo la cantidad de errores puede ser muy alta y acabar siendo muy baja

- Por esto, necesitamos método más potente
IP3 algorithm

- Sorpresa de la observación:

$$= -\log_2 (P(V=v))$$

- entropía: valor esperado de la sorpresa

$$\begin{cases} =1 & \Rightarrow \text{desigualdad } 50, 50 \\ =0 & \Rightarrow \text{todos los elementos son de 1 clase} \end{cases}$$

$$H(v) = - \sum_v P(V=v) \cdot \log_2 (P(V=v))$$

gir
9 positivos y 5 negativos

$$-(9/14) \cdot \log_2 (9/14) - (5/14) \cdot \log_2 (5/14) = 0.94$$

- Luego debo ponderar las entropías en función del número de ejemplos de cada una

- Calculo la entropía de cada característica y a continuación, la ganancia

- Luego sigo mirando los de abajo

- Cuando tengamos clasificados todos los ejemplos, podemos saber por qué hemos llegado a esa conclusión
controles!

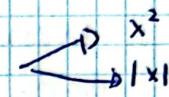
- Sobreaprendizaje

- Debemos utilizar un criterio distinto para poder (mse)

Tema 11

Algoritmos genéticos (Heurístico)

- Es un proceso de optimización, utilizado para encontrar parámetros óptimos en cualquier modelo
- Cuenta + rápido, + fácil
- Se puede paralelizar
- La codificación es esencial (Materiales de un helicóptero)
- Lo difícil es traducir algo natural a cadena de bits
- Es muy sencillo cambiar algunos parámetros objetivos.



- son procedimientos heurísticos y estocásticos
- Cada vez que se ejecutan, dan un resultado distinto

- Se parte de una población inicial de candidatos
- Cada uno de estos y sus siguientes son evaluados con una función de aptitud (Fitness Function)

↳ Representa que tan buena es la solución

(que tanto se adapta el individuo al entorno)

- La función Fitness se evalúa según el fenotípico del individuo.
- (las características observables, cultura)
- La genética tiene que ver con el genotípico, es la forma en que se codifica genéticamente el espécimen
- El proceso de transformar el fenotípico en el genotípico es: encoding

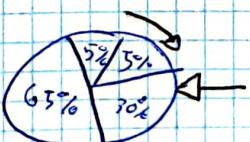
Operadores Genéticos

Selección

- Debemos buscar el que tenga un Fitness más alto, ya que será el que esté más adaptado al medio.

$$P(\text{Individuo}(i)) = \frac{\text{Fitness}(\text{Individuo}(i))}{\sum_{i=1}^n \text{Fitness}(\text{Individuo}(i))}$$

- La implementación previa a la selección se llama "ruleta de ruleta"



Madre: 8 (65%) 001000
Padre: 3 (9%) 000011

• El siguiente operador básico es el cruce o recombinación

Madre 001000

Padre 000011

hijo 001011

• Mutación

Cambiar aleatoriamente una de las posiciones

Antes 001011

Después 001~~0~~11

{

- Fitness → Función objetivo
- Individuo → Solución
- Generación → Población
- Fenotipo → Solución decodificada
- Genotípo → Solución codificada
- Gene → String binaria

Modificaciones

② Inversión 0111~~001~~010
0111 100 010

③ Cruce uniforme

(selecciona unos puntos concretos de cada uno)

Madre 100101001000101

Padre 111000110111111

Template 10110011000111

④ Cruce multi punto

Madre 001000

Padre 100011

Offspring 1001011

Offspring 2 100000

① Algoritmos híbridos

- Consiste en combinar otros algoritmos

Otros algoritmos de evolución

• Estrategias evolutivas

- Reproducciónsexual

◦ (λ, λ)

– 1 parente genera λ hijos

– Los hijos son mutaciones del parente

– Paramos cuando → NO, tengamos una mejora significativa
↳ Si los individuos son muy muy parecidos

– Si tengo muchos hijos mejores que el parente, es porque estoy cambiando demasiado el parente

– Para solucionarlo, calculo en los últimos generaciones, si es > al 20%
↳ tengo que reducir la mutación (regla 1/5)

◦ (μ, λ)

◦ $(\mu + \lambda)$

• Programación genética

- Máquinas programando

Tema 10

Aprendí zojo bayesiano

- Yo asigno una probabilidad a un suceso en función de mi cerebro pero luego al ver los datos me darán la razón o no

Ej:

¿Cuántos días va a llover este año?

70% porque lo digo yo, y luego los datos me harán cambiar de opinión

- Tiene la ventaja que necesita muy pocos datos

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)}$$

- Si es difícil calcularlo de esta forma, puede que si el cambio de llegar los datos sea más fácil

- Si no sabemos los porcentajes de un suceso, puedo aplicar la distribución previa y dar 50% 50%

ej:

$$P(\theta|x) = \frac{P(x|\theta) \cdot P(\theta)}{P(x)}$$

Partido fútbol

$$\left\{ \begin{array}{l} P(\theta|x) = \text{Probabilidad a posteriori} \\ P(x) = \text{Distribución marginal de los datos} \\ P(\theta) = \text{Probabilidad a priori} \\ P(x|\theta) = \text{Función de verosimilitud} \end{array} \right.$$

- Como P(x) suele ser difícil de calcular, paso de ello

- No nos interesa un valor en concreto, sino si con un conjunto de datos da una mayor probabilidad que con otro

- Es común calcular los valores de los parámetros que maximicen la posterior distribución.

Procedimiento máximo a posteriori (MAP)

$$\Theta_{MAP} = \arg \max P(x|\theta) \cdot P(\theta)$$

(La suma total no tiene porque dar 1)

Naïve bayes

$$P(X_1, X_2, \dots, X_m | Y) = \prod_{i=1}^m p(X_i | Y)$$

Punto:

long	sweet	yellow	fruit	TOTAL
400	350	450	Banana	500
0	250	300	Orange	300
200	150	30	other	<u>200</u> 1000

- Encontramos una long, amarilla y sweet ó fruta?

- $P(\text{banana} | \text{long, sweet, yellow}) =$

$$= p(\text{long} | \text{Banana}) \cdot p(\text{sweet} | \text{Banana}) \cdot p(\text{yellow} | \text{Banana}) \cdot p(\text{banana}) =$$

$$= \frac{400}{500} \cdot \frac{350}{500} \cdot \frac{450}{500} \cdot \frac{500}{1000} = 0.12688$$

- $P(\text{orange} | \text{long, sweet, yellow}) = 0$

- $P(\text{other} | \text{long, sweet, yellow}) = 0.0027$

- Lo calculo para todos y me quedo con el de probabilidad mayor (esto reduce mucho la carga computacional)

- tenemos el problema de que si suma es 0, todo va a ser 0

→

aplico la conexión Laplaceana

- Me creo unos ejemplos inventados pero de forma justa

Ventajas

- Fácil de entender
- Aplicable con bases de datos pequeñas
- Si se cumple la independencia condicional, este es el mejor modelo
- Excelente rendimiento
- Muy eficiente computacionalmente
- Insensitivo a outliers
- Fácil de incorporar nuevos datos

Desventajas

- Probabilidades con ceros, requieren modificaciones artificiales
- No proporciona probabilidades "reales"
- No se puede utilizar en problemas de regresión
- No aprende interacciones entre funciones
- Rendimiento bajo con alta interacción entre las variables

Introducción al aprendizaje profundo

- modelos Deep Learning
 - En algunos diagnósticos es mejor que los expertos del tema
 - Utillizados en los sistemas de desarrollo (completar el proteoma humano)
- El deep learning es solo una forma de llamar a los modelos de Redes Neuronales Artificiales (ANN)
- ANN simulan el cerebro animal y también su estructura
- Algoritmos genéticos y ANN simulan las capacidades computacionales biológicas
 - (Muy sensible a los hiperparámetros)
- La unidad básica de procesamiento de info es la Neurona
(Ramon y Cajal y Golgi XIX)
- Neuronas intercambian info a través de la Sinapsis, señales electroquímicas que se propagan desde el círculo sináptico a través de las Dendritas hacia el Soma, (cuerpo de la célula)
- Cuando se alcanza un umbral, la célula libera una señal de activación a través de su axón hacia las neuronas vecinas
- En ANN podemos llamar a las neuronas como modos
- Los modos se organizan en capas (reciben, transforman y transfieren señales ponderadas a la siguiente capa)
- La Capa de entrada recibe info del exterior
 - Estas entradas se transforman a través de una Función de activación (generalmente no es lineal) y se pasan a otros modos multiplicándolos previamente por un Peso
- Se repite el proceso hasta llegar a la Capa de salida
- El proceso de modificar la estructura hasta obtener la salida esperada se llama Aprendizaje

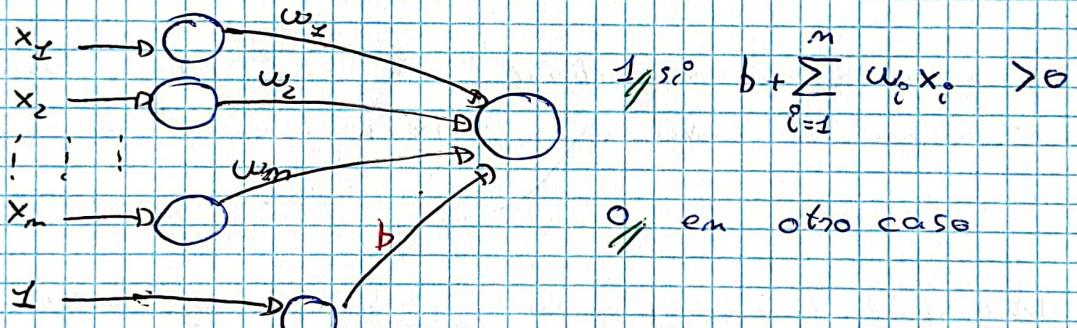
- Las capas que se encuentran entre las de entrada y salida, se llaman **Capas ocultas**

- Estas arquitecturas se les llama **Redes Prealimentadas multicapa** o también **Perceptrones multicapa** (forma no lineal)
- (Aún es análogo al Deep Learning)

Perceptron:

- Formado por una capa de neuronas de entrada y una capa de salida (que produce 0 o 1)
- (PSe alcanza o no un umbral)

- Las entradas $x_i = 1, 2, 3, \dots, m$ se conectan con unos pesos ponderados, **Pesos** w_1, w_2, \dots, w_m , a la unidad de salida (entendido por peso, $x_i w_i$)
- Se utiliza un peso especial, b , llamado **Sesgo** y se conecta con un modo hipotético, con valor siempre 1



• el ejemplo de XOR es imposible

ya que el perceptrón solo funciona para funciones separables linealmente

- Para simplificar

$$b = w_0 \quad y \quad x_0 = 1$$

- Para problemas linealmente separables, podemos emplear el algoritmo del perceptrón

- El aprendizaje es exactamente encontrar los pesos óptimos para realizar la tarea

c) b • Ponemos a 0 todos los sesgos

$$w_0 = w_1 = w_2 = \dots = w_m = 0$$

• Para m iteraciones:

Si $\phi_{bw}(x^k) = \underbrace{a^k}_{\text{Varios pesos esperados}}$ para todo $k \Rightarrow$ para

símo, para todo k , modifiquemos todos los pesos

$$\Delta w_i = (a^k - \phi_{bw}(x^k)) \circ x_i^k$$

ej

XNOR

x_1	x_2	y
1	1	1
0	0	1
1	0	0
0	1	0

Selecciónamos estas entradas

Recuerda que $x_0 = 1$

$$\text{tenemos } X^3 = (x_0^3, x_1^3, x_2^3) = (1 \ 1 \ 0)$$

e

Iniciaremos el sesgo a $0 = w_0 = w_1 = w_2$

La salida de ejemplo será:

$$\phi_w(x^3) = (w_0 x_0 + w_1 x_1 + w_2 x_2) = \phi(0 \cdot 1 + 0 \cdot 1 + 0 \cdot 0) = 0$$

Dado que $\phi(x^3) = 0 \Rightarrow$ no tengo que cambiar los pesos
Lo esperado

c) c

XNOR

x_1	x_2	y
0	0	1

$$x^3 = (1, 0, 0) \quad \text{y sesgo parte de } w_0 = w_1 = w_2 = 0$$

$$\phi_w(x^3) = \phi(0 \cdot 1 + 0 \cdot 0 + 0 \cdot 0) = 0 \rightarrow \text{pero yo espero } 1$$

¶

$$\Delta w_0 = (a^3 - \phi_w(x^3)) x_0^3 = (1 - 0) \circ 1 = 1$$

$$\Delta w_1 = (a^3 - \phi_w(x^3)) x_1^3 = (1 - 0) \circ 0 = 0$$

$$\Delta w_2 = (a^3 - \phi_w(x^3)) x_2^3 = (1 - 0) \circ 0 = 0$$

• Cambio los pesos

$$w'_0 = w_0 + \Delta w_0 = 0 + 1 = 1$$

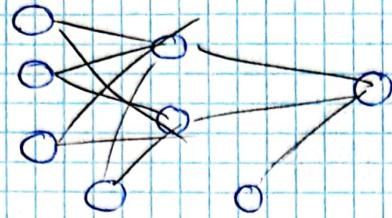
$$w'_1 = w_1 + \Delta w_1 = 0 + 0 = 0$$

$$w'_2 = w_2 + \Delta w_2 = 0 + 0 = 0$$

Iteramos igual hasta el resultado

esperado

Redes neuronales prealmacenadas



- A las capas que se encuentran entre la de entrada y salida se llaman Capas ocultas
- El número de unidades en cada capa puede variar
- No existen conexiones hacia atrás o bucles
- Más de 1 Salida
- Ahora las neuronas ejecutarán alguna función, a esto le llamaremos función de activación.
- Intentan emular algunas transformaciones de los señales entrantes
- Podemos utilizar cualquier función no lineal, pero utilizaremos función Sigmoidal (función logística)

$$f(x) = \frac{1}{1 + e^{-x}}$$



- Valor mínimo es 0 y máximo es 1
- $[0, 1]$ cuando antes era solo $\{0, 1\}$
- Cada vez que ejecutemos la red neuronal, nos dará un resultado distinto
- Aproximación universal
- La ventaja de estas redes frente al perceptrón es:
- * {Com una sola capa intermedia y un número suficiente de neuronas intermedias y una función de activación, es capaz de aprender cualquier función.

(Demostrado por Hornik)

- No sabemos cuantas neuronas son suficientes

$$Y = f(x) \approx \phi(X, \Omega, \Theta)$$

- Las redes alimentadas hacia adelante tiene el problema de encontrar los pesos, se puede decir cuáles son los pesos pero no como encontrarlos
 - gracias al algoritmo de retropropagación somos capaces de encontrar los pesos
- Las redes alimentadas hacia adelante, frente a los perceptrones, tienen un coste:
 - La dimensionalidad es mucho mayor y complica enormemente el proceso de aprendizaje
- Entradas: r
- Unidades ocultas: q
- Salidas: s
- (Los pesos para las unidades de salida) = $q(r + s + 2)$
- Aprendizaje
 - { - Se presenta alguna entrada a la red
 - Se realiza una pasada hacia delante para calcular la salida
 - El error de salida se calcula comparando las salidas reales y las esperadas ($y' - y$)
 - Dicho error se utiliza para modificar los pesos correspondientes.
- Se necesitan varias pasadas por el conjunto de entrenamiento
- Un tema crucial en este tipo de redes es la elección de la función de pérdida
 - En perceptrón no la definímos ya que solo tenía 2 clases posibles
- En las redes alimentadas hacia delante, es necesario definir la función de pérdida específica para el problema.
 - De acuerdo con esta función, escribiremos el algoritmo correspondiente para optimizarlo.
(Esto puede ser engorroso)
- No es lo mismo predecir que va a tener cáncer a que no va a tener cáncer

Modificaciones redes preimplementadas:

- Funciones de activación no lineales
- Función de pérdida arbitraria
- Múltiples capas ocultas

- El cambio de peso en una neurona no tiene porque producir un cambio en la salida



- En consecuencia, necesitamos evaluar el efecto de error en la función de pérdida

$$\frac{\text{cambio en error}}{\text{cambio en un peso}}$$

- Se representa:

$$\frac{\partial L}{\partial w_j} \quad L = \text{función de pérdida}$$

$w_j = \text{cualquiera de los pesos de la red}$

- En función del problema, utiliza raras una función de pérdida

- En clasificación binaria, entropía negativa

- Para poder cambiar los pesos y hacer disminuir el error, se debe calcular la derivada parcial de todos los pesos

- Si la pérdida aumenta cuando aumentamos el peso, debemos disminuir el peso

- Los pesos deben cambiarse moviéndose en la dirección opuesta a la derivada parcial.

• Algoritmo de retropropagación

- (cuando existe n capas ocultas en una red, no sabemos cuál es la salida esperada de una de estos neuronas entonces no podemos calcular el cambio de este para que mejore)

- Se puede calcular el error (pérdida) en la salida y luego atribuir parte del error a cada una de las unidades ocultas dependiendo de su participación.

- Proceso

- Se ejecuta el algoritmo hacia delante, fase de **avance**.
- Luego el error de salida se atribuye a cada una de las unidades ocultas considerando su participación, modificamos los pesos, **fase hacia atrás**.
- El cálculo de los errores atribuidos se hace con la **regla de la cadena**.
- Este algoritmo se ejecuta para todos los ejemplos.

Hiperparámetros

- N° capas ocultas
- N° de neuronas
- Momento
- Tasa de aprendizaje