Aprendizaje supervisado

Ventajas Son razonablemente resistenteas al fallo, son capaces de "comerse" o filtrar errores moderados. Son altamente escalables, las entradas pueden tener una cantidad nmuy amplia de entradas, con añadir mas neuronas o capas (Deep Learning) crecen y se adaptan (No hace falta cambiar el codigo mucho) Pueden resolver problemas no lineales, es decir, que los datos de modelo generado no tienen porque separables en base a los datos de entrada.

Desventajas No son modelos explicables, son cajas negras entrenadas pero no cononocemos como funcionan internamente (Sin intelequias). Se está trabajando en ello, pero actualmente no hay formas de coger la red e interpretarla. Es problemático éticamente usar modelos sin comprenderlos pore ejemplo para usos en una central nucelar, un analisis de un tumor para decidir si usar quimio.... El alto coste computacional restringe las aplicaciones en base a las restricciones de ahrdware, por ejemplo en una raspberri pi o en lugares con mala conexion donde no podemos acceder a una red computacional remota. Una red neuronal corriendo en un rover en marte puede ser vital cuando una amenaza se dirige proxima y el retardo de varios minutos desde la tierra es insalvable.

5.2 Tecnologia KNN

K vecinos más cercanos. Cuando se entrena no se analizan los datos, se almacenan (en una lista por ejemplo) para mas tarde ante un caso nuevo, se comparan los almacenados y se elige de la lista el más cercano. Esto significa que si dos ejemplos son parecidos se asume que la respuesta es parecida. Si dos personas son semejantes en vestimenta y en gustos musicales, puedes inferir nuevas canciones que puedan gustarles. Se usa en recomendadores, se comparan gustos entre usuarios de caracteristicas semejantes (edad, sexo, peliculas, etc) y se cruzan las recomendaciones entre ellos. No actua TANTO como caja negra porque podemos explicar la decision usando los ejemplos de entrenamiento. La K es porque a veces no se elije solo uno parecido sino k elementos antes de tomar la decisión.

> El tiempo de entrenamiento es 0 Se suelen llamar técnicas perezosas. El tiempo de recuperación de resultado depende de la base de ejemplos. Podemos usar como medida:
> Distancias Euclidea

$D-Euclidea(X,Y)=\sqrt[2]{\sum^N_{i=1}(x_i-y_i)^2}$

Distancia Manhattan:

$D-Manhattan(X,Y)=\sum_{i=1}^{N} |x_i-y_i|$

Por ejemplo con un algoritmo en Tinder

Altura	Peso	ColorPelo	ColorPiel	Genero	Match
1.80	90	R	B	M	V
1.70	70	M	M	M	X

Dado el caso de una persona calculamos las diferencias con los ejemplos que tenemos

1.75	75	PR	B	M	?
0.05	15	1	0	1	?
0.05	5	1	1	1	?

En principio se parece mas al segundo ejemplo asique el Match daría negativo

Pdoemos empezar a ver varios problemas, codificaciond de categorias (Color de piel, pelo, genero....) Estamos asumiendo que una diferencia te altura es mas importante que una de color de pelo (Valor 5 respecto a valor 1)

> Es importante que los atributos estén en la misma escala (Normalizarlos) Son todos los valores igual de immportantes para el caso?

Distancia de Minkowsi:

$D-Minkowsi(X,Y)=\sqrt[p]{\sum^N_{i=1}(x_i-y_i)^p}$

Distancia de Levenstein: -Funciona bien en palabras -Ignora relaciones diagonales en los datos

C	a	s	a
c	a	l	a
0	0	1	0

Solo hay 1 cambio de letra

C	a	s	a	
c	a	l	l	e
0	0	1	1	?

2 cambios y 1 inserción

Esta no cae en el examen (ignorar código)

Distancia de Mahalanobis:

$D-Mahalanobis(\vec{x},\vec{y})=\sqrt[2]{(x-\mu)^TC-1(x-\mu)}$

Donde μ es la media de los datos y $C-1$ la inversa de la matriz de covarianza

Podemos introducir pesos diferentes a las variables $Weighted Euclidean Distance(X,Y,P)=\sqrt[2]{\sum^{N}_{i=1}(x_i-y_i)^2}$

5.2.1 Ventajas de KNN

1. NO parametrico, no asume nada de los datos porque los usa como ejemplos
2. Simple de explicar porque no actua con funcionamiento interno
3. Buena precisión, siempre que el modelo sea compatible con KNN
4. El entrenamiento es instantáneo

5.2.1 Desventajas de KNN

1. Sensible a valores irrelevantes, hay que decidirlos bien. SI en el ejemplo estamos metiendo como dato el color de piel, pero es un dato irrelevante, afectará a la similitud pese a ser irrelevante.
2. Sensible al ruido, los malos ejemplos pueden dar malos.resultados (Mitigado por k)
3. Si el volumend e datos es grande la ejecución es lenta. Por ejemplo teniendos digitalizados todos los pacientes de un hospital. Suele usarse CPU
4. Es costoso en memoria.

> Naughty Dog suele usar este algoritmo para decidir las animaciones en vez de un arbol de estados para relacionarlas, haciendo adaptaciones psoteriores

Ejemplo de KNN en SKLearn

> El OCR no va bien, me toca hacer el codigo a mano cuando acabe ;_;

```
iris . data))
X train
iris.data[: -20]
y _ train = iris.
X test = iris. data[-20:]
y _ test: iris. target[-2ô:]
from sklearn. neighbors import KNeighborsClassifier
model
model.fit(X_train, y_train)
prediction =
print("Predicción")
print(prediction)
print("Test")
print(y_test)
model. score (X_test,
y _ test)
```

5.3 CBR

Se diferencia de la KNN por la idnexaciond e casos, son soluciones adaptables. Se actualiza la base de casos con los casos psoteriores. Se suelen depurar los casos manualmente, eliminar los no usados, los mas antiguos...

5.3.1 Similaridad

5.3.2 Señor va putorapido, lo edito luego

Y eso que estoy haciendo la mierda de latex con las formulas, sheesh

5.3.3 El profe se ha saltado un numero, alleva un kirbo

5.3.4 Aprendizaje

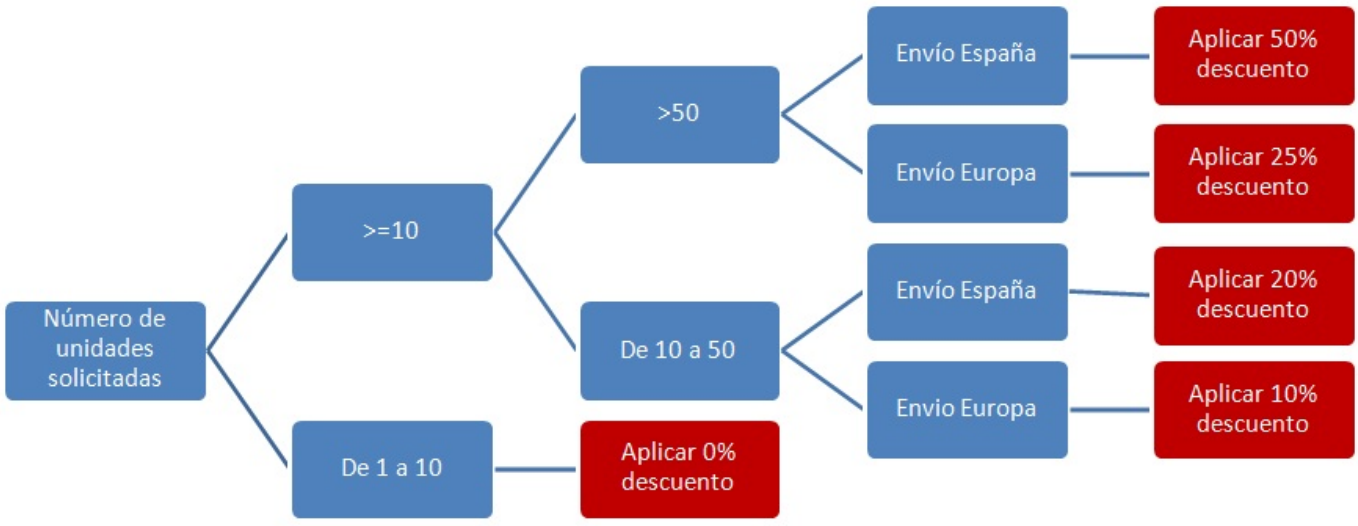
Se supervisa la adaptacion del nuevo caso: el mecánico aprueba

5.3.5 Otras medidas

No entra en examen, pero hay formulicas de google La siguiente pagina es la simplificacion de la formula Veces de termino partido importance Su importancia es las veces del termino en la cantidad de documentos

5.4 Arbol de decisiones

Usa nodos condicionales



El algoritmo usa entropia para ayudar a las decisiones

$Entropy(s)=\sum^c_{n=1}-p_i*Log_2p_i$

Ganancia:

$Gain(S,A)=Entropy(S)-\sum^{|A|}_{v\forall V(A)}{\frac{|sv|}{s}Entropy(S_v)}$

En este ejemplo ha elegido el atributo de unidades compradas porque se ha calculado como valor de entropia que mejor discrimina los datos. Realmente actua como una busqueda voraz.

> "Imaginemos una vaca esférica" ~Zero, se aburre

Sitio de acceso A1	Cantidad gastada A2	Vivienda A3	Última compra A4	Clase
1	0	2	Book	Good
1	0	1	Disc	Bad
1	2	0	Book	Good
0	2	1	Book	Good
1	1	1	Book	Bad
2	2	1	Book	bad

Elegimos el mejor atributo donde I_{ij} son los posibles valores de I, {0,1,2}
$A_1=P*I_{10}*+P*I_{11}+P*I_{12}\newline \qquad\frac{1}{6}*I_{10}+\frac{4}{6}*I_{11}+\frac{1}{6}*I_{12}$

Calculamos la entropia de cada valor I_{ij}

Donde las fracciones representan la cantidad de valores que clasifican como good menos los que clasifican como bad:

$I_{10}=-\frac{1}{1}log_2\frac{1}{1}-\frac{0}{1}log_2\frac{0}{1}=-1*0+0=0$

$I_{11}=-\frac{2}{4}log_2\frac{2}{4}-\frac{2}{4}log_2\frac{2}{4}$

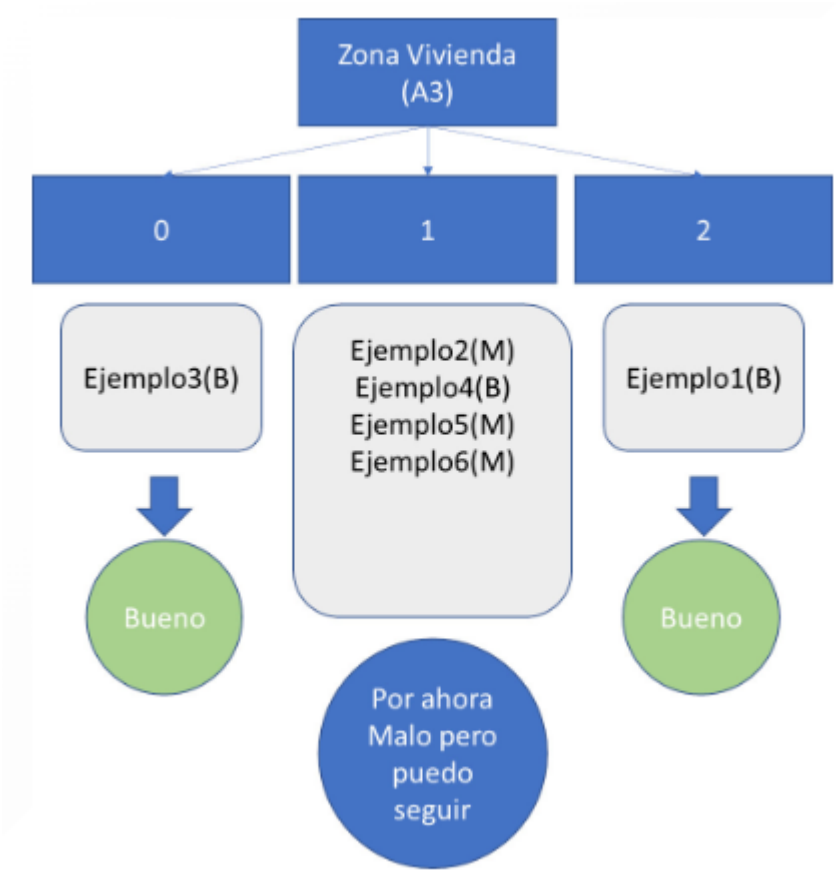
$I_{12}=0$ porque hay 0 buenos y 1 malo

$\boxed{Log_a b = \frac{log_{x}b}{log_{x}a}}$

Si seguimos haciendo esto con los atributos el que devuelve el valor de entropia mas bajo es A3

$\begin{cases}A1=0,66\A2=0,79\A3=0,54\A4=0,81\end{cases}$

Esto significa que si se usa como primer atributo va a separar la mayor cantidad de clases


Zona Vivienda (A3)
0
1
2
Ejemplo3(B)
Ejemplo2(M)
Ejemplo4(B)
Ejemplo5(M)
Ejemplo6(M)
Ejemplo1(B)
Bueno
Bueno
Por ahora Malo pero puedo seguir