APELLIDOS: PL:

NOMBRE: DNI:

ESCUELA DE INGENIERÍA INGORMÁTICA

SISTEMAS INTELIGENTES

Examen Final de Teoría. Viernes 11 de enero de 2019.

I.- Búsqueda (3,5 puntos)

1.- Se trata de resolver, mediante búsqueda en espacios de estados, con A* por supuesto, una variante del problema del 8-puzzle en la que en lugar de tener 8 fichas distintas (1,2,3,4,5,6,7,8), tenemos 8 fichas tales que son iguales dos a dos (1,1,2,2,3,3,4,4). El estado inicial puede ser cualquier distribución de las fichas en el tablero, por ejemplo:

$$\begin{bmatrix} 3 & 3 & 4 \\ 4 & & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

Y se trata de llegar a una configuración en la que la fichas estén ordenadas de la forma siguiente:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

Los movimientos permitidos son los mismos que en el 8-puzzle normal, es decir se puede mover una ficha de su posición a la posición vacía si las dos posiciones son adyacentes ortogonalmente. El objetivo es también calcular la secuencia mínima de movimientos para llegar de la configuración inicial al objetivo.
Lo que se pide en este ejercicio es modelar el espacio de búsqueda y definir buenos heurísticos, **concretamente se pide responder de forma razonada a las siguientes cuestiones**:

a) **[0,5 puntos]** ¿Cómo son los estados, las reglas y sus costes? Dibujar la parte del estado de búsqueda correspondiente al estado inicial anterior y sus sucesores.

Solución: Esto es como en el 8-puzzle convencional: Los estados son configuraciones válidas del tablero, por lo tanto, se pueden representar mediante permutaciones de las 8 fichas y un 0 para representar a la posición vacía. Por ejemplo, los estados anteriores estarían representados por las permutaciones (3 3 4 4 0 1 2 1 2) y (1 2 3 4 0 1 2 3 4) respectivamente.
Las reglas aplicables a un estado vienen dadas por todas las posibilidades de mover una ficha a la posición vacía, si esta posición es adyacente ortogonalmente a la de la ficha; y los costes de todas las reglas son 1 dado que lo que se pretende en minimizar el número de movimientos.
Con todo esto, los sucesores del estado inicial anterior son los siguientes

$$\begin{bmatrix} 3 & 3 & 4 \\ 4 & & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 3 & & 4 \\ 4 & 3 & 1 \\ 2 & 1 & 2 \end{bmatrix} \quad \begin{bmatrix} 3 & 3 & 4 \\ & 4 & 1 \\ 2 & 1 & 2 \end{bmatrix} \quad \begin{bmatrix} 3 & 3 & 4 \\ 4 & 1 & \\ 2 & 1 & 2 \end{bmatrix} \quad \begin{bmatrix} 3 & 3 & 4 \\ 4 & 1 & 1 \\ 2 & & 2 \end{bmatrix}$$

b) **[0,25 puntos]** Dar una cota superior, lo más ajustada posible, del número de estados del espacio de búsqueda.

Solución: En principio el número de estados diferentes es el número de permutaciones diferentes que se pueden generar con las 8 fichas y el 0, es decir 9!. Pero en este caso como las 8 fichas son iguales 2 a 2, tendremos que dividir este número 4 veces entre 2!. Es decir la cota que podemos dar es 9!/(4*2!) = 181440/8 = 22.680 estados diferentes. Dado un estado inicial cualquiera, estos son en principio los estados que serían alcanzables desde el estado inicial. Un estudio más detallado nos permitiría, posiblemente, reducir este número de forma similar a cómo en el 8-puzzle normal se reduce a 9!/2, pero esto va más allá de lo que hay que hacer en esta pregunta.

c) **[1,25 puntos]** Consideremos el heurístico $h_a(n)$ definido de forma que calcula para cada ficha la distancia ortogonal desde la posición de la ficha en el estado n a la posición más cercana en la que podría estar esa ficha en el objetivo, y luego suma todas estas distancias. Indicar el valor de este heurístico para todos los nodos dibujados en a). ¿Está h_a bien definido? ¿Es h_a admisible? ¿Es h_a monótono?

Solución: El valor del heurístico h_a para los 5 nodos de la figura anterior es (recorriendo las fichas por filas y los nodos sucesores de izquierda a derecha):
$h_a(\text{inicial}) = 2+1+1+0+1+1+1+1 = 8$.
$h_a(n_1) = 2+2+1+0+1+1+1+1 = 9$.
$h_a(n_2) = 2+1+1+1+1+1+1+1 = 9$.
$h_a(n_3) = 2+1+1+0+2+1+1+1 = 9$.
$h_a(n_4) = 2+1+1+0+1+1+2+1 = 9$.

El heurístico $h_a(.)$ está bien definido ya que $h_a(n) \geq 0$ para todo nodo n, y $h_a(\text{objetivo}) = 0$.
También es claro que es admisible, es decir $h_a(n) \leq h^*(n)$ para todo nodo n, dado que para llevar una ficha a su posición en el objetivo como mínimo hay que moverla tantas veces como su distancia ortogonal a la posición más cercana a la posición actual que puede ocupar esa ficha en el objetivo.
Para demostrar que es monótono (o consistente que es equivalente), es decir que $h_a(n_1) \leq h_a(n_2) + c(n_1,n_2)$ para todo par de nodos n_1,n_2, basta con tener en cuenta que el cambio en el valor del heurístico al pasar de un estado a un sucesor es +1, 0 ó -1, y que $c(n_1,n_2)$ es 1 si n_2 es sucesor de n_1 e ∞ en otro caso.

d) **[1 punto]** Consideremos ahora la siguiente relajación del problema: eliminamos la restricción de que no puede haber dos fichas en la misma posición, es decir se puede mover una ficha a una posición adyacente ortogonalmente aunque esta posición no esté vacía. ¿Cómo es el heurístico h_b resultante de esta relajación? Dar el valor de h_b para el estado inicial. Pista: puede ser de utilidad definir funciones: $D_{11}(n,i)$ = distancia ortogonal de la primera aparición de la ficha i en el estado n (siguiendo el orden de las filas) hasta la posición de la primera aparición de i en el objetivo. Análogamente D_{12}, D_{21} y D_{22}.

Solución: Si eliminamos la restricción de que en una casilla solamente puede haber una ficha, entonces para resolver de forma óptima el problema podríamos mover las fichas siguiendo un camino ortogonal desde su posición actual hasta una de las posiciones que cada ficha puede ocupar en el objetivo. Pero para cada par de fichas iguales, tenemos que tener en cuenta que no pueden ir las dos a la misma posición, por eso hay que considerar para cada par de fichas dos opciones y quedarnos con la que produzca un menor coste. El coste de resolver este problema relajado para un nodo n (y por lo tanto el valor del heurístico h_b) se puede calcular con ayuda de las funciones anteriores como:

$$h_b(n) = \sum_{i=1}^{4} min\big(D_{11}(n,i) + D_{22}(n,i), D_{12}(n,i) + D_{21}(n,i)\big)$$

El valor de este heurístico para los estados anteriores es (considerando el orden de las fichas 1, 2, 3, 4):
h_b(inicial) = min(1+3, 3+1) + min(1+3, 3+1) + min(2+1, 2+3) + min(1+0, 3+2) = 12. $h_b(n_1)$ = min(1+3, 3+1) + min(1+3, 3+1) + min(2+2, 2+2) + min(1+0, 3+2) = 13.
$h_b(n_2)$ = min(1+3, 3+1) + min(1+3, 3+1) + min(2+1, 2+3) + min(1+1, 3+1) = 13.
$h_b(n_3)$ = min(2+1, 3+2) + min(1+3, 3+1) + min(2+1, 2+3) + min(1+0, 3+2) = 11.
$h_b(n_4)$ = min(2+1, 3+2) + min(1+3, 3+1) + min(2+1, 2+3) + min(1+0, 3+2) = 11.

e) **[0,5 puntos]** ¿Es h_b consistente? ¿Se puede afirmar que h_b domina ampliamente a h_a?

Solución: El heurístico h_b es consistente dado que se ha definido mediante el método de la relajación del problema (y ocurre que todos los heurísticos definidos mediante este método son consistentes). Además, ocurre que para todo nodo n $h_b(n) \geq h_a(n)$; esto es claro dado que para resolver el problema relajado que da lugar al heurístico h_b no se pueden mover las dos fichas iguales a la misma posición, y la suma de distancias ortogonales de cada par de fichas iguales a la posición más cercana en el objetivo al que puede ir es menor que la suma de los movimientos elementales que tendríamos que hacer para llevar las dos fichas a sus posiciones en el objetivo de la forma menos costosa posible. Por lo tanto h_b domina ampliamente a h_a.

II.- Representación (3 puntos)

3.- **[1.25 punto]** Se desea implementar un sistema de reglas (un bot) para resolver un pequeño laberinto, el cual cuenta con distintas habitaciones comunicadas mediante puertas. No obstante, algunas puertas tienen cerrojo y requieren una llave para poder abrirse. El dibujo muestra un posible laberinto.

Diseñar un sistema de reglas en CLIPS tal que:

- El jugador se desplace de una habitación a otra si hay una puerta cerrada entre ellas y tiene la llave. Una vez abierta la puerta, se queda abierta para siempre.
- Si no hay puertas cerradas en la habitación, o el jugador no tiene la llave, entonces que cruce una de las puertas abiertas (si hay).
- El jugador no vuelva nunca a la habitación inmediatamente anterior (evitar bucles infinitos)
- Si el jugador se encuentra en la habitación que tiene la llave, que la recoja (el jugador pasa a tener la llave, y ésta desaparece de donde esté).
- El juego finaliza cuando el jugador llega a la habitación llamada "salida", en cuyo caso se mostrará un mensaje indicando que ha ganado.

Para ello pueden utilizarse las siguientes plantillas de hechos en CLIPS:

```
(deftemplate puerta
	(slot origen) ; Habitación origen
	(slot destino) ; Habitación destino
```

```
	(slot abierta (allowed-symbols no si)) ; Indica si la puerta está abierta o requiere llave
)
(deftemplate jugador
	(slot habitacion) ; Habitación actual
	(slot tieneLlave (allowed-symbols no si)) ; Indica si tiene la llave o no
	(slot ultimaHab) ; Última habitación en la que ha estado
)
(deftemplate llave
	(slot habitacion) ; habitación donde se encuentra la llave
)
```

Solución:

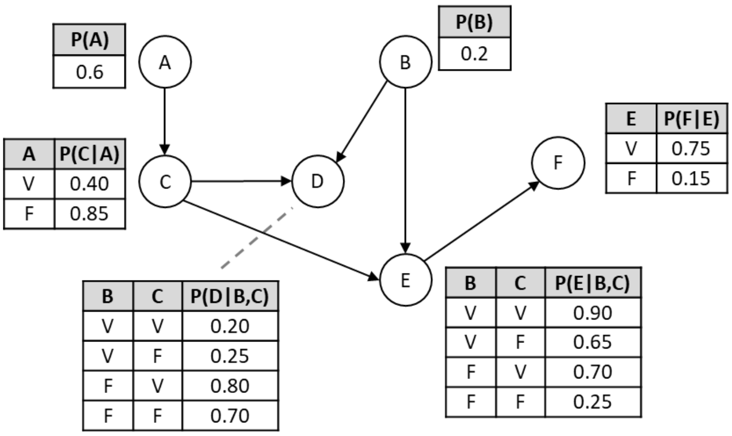
```
(defrule cruzar-cerrada
	?j <- (jugador (habitacion ?h) (tieneLlave si) (ultimaHab ?u))
	?p <- (puerta (origen ?h) (destino ?d & ~?u) (abierta no))
	=>
	(assert (jugador (habitacion ?d) (tieneLlave si) (ultimaHab ?h)))
	(assert (puerta (origen ?h) (destino ?d) (abierta si)))
	(retract ?j)
	(retract ?p))

(defrule cruzar-abierta
	(declare (salience -1))
	?j <- (jugador (habitacion ?h) (tieneLlave ?ll) (ultimaHab ?u))
	(puerta (origen ?h) (destino ?d & ~?u) (abierta si))
	=>
	(assert (jugador (habitacion ?d) (tieneLlave ?ll) (ultimaHab ?h)))
	(retract ?j))

(defrule coger-llave
	?j <- (jugador (habitacion ?h) (tieneLlave no) (ultimaHab ?u))
	?ll <- (llave (habitacion ?h))
	=>
	(assert (jugador (habitacion ?h) (tieneLlave si) (ultimaHab ?u)))
	(retract ?j)
	(retract ?ll))

(defrule ganar
	?j <- (jugador (habitacion salida))
	=>
	(printout t "Has ganado" crlf)
	(retract ?j))
```

4.- Dada la siguiente red bayesiana, responder razonadamente a las siguientes preguntas:



P(A)
0.6

P(B)
0.2

A	P(C|A)
V	0.40
F	0.85

E	P(F|E)
V	0.75
F	0.15

B	C	P(D|B,C)
V	V	0.20
V	F	0.25
F	V	0.80
F	F	0.70

B	C	P(E|B,C)
V	V	0.90
V	F	0.65
F	V	0.70
F	F	0.25

a) **[0,75 puntos]** ¿Son las variables aleatorias A y B independientes? ¿Y las variables F y A? Justifica ambas respuestas utilizando el criterio de D-separación.

Solución: Existen dos caminos distintos entre las variables A y B.

Camino 1: A -> C -> D <- B

Camino 2: A -> C -> E <- B

En el primer camino existe un bloqueo en el nodo D, ya que es un nodo cabeza-cabeza no observado y sin descendientes. En el segundo camino, el bloqueo lo produce el nodo E al ser un nodo cabeza-cabeza y no estar observado ni él ni su descendiente F. Dado que todos los caminos entre A y B están bloqueados, **A y B son independientes.**

En el caso de las variables A y F hay de nuevo dos caminos:

Camino 1: A -> C -> E -> F

Camino 2: A -> C -> D <- B -> E -> F

En el primer camino no existe ningún bloqueo, ya que todos los nodos intermedios son del tipo no_cabeza-cabeza y no están observados. Dado que existe un camino sin bloqueos, podemos afirmar que **A y F <u>no</u> son independientes**.

b) **[0,75 puntos]** Sabiendo que A y F se cumplen, pero B y E no, ¿cuál es la probabilidad de que se cumpla D?

Solución:

$$P(D|A,\neg B,\neg E,F) = \frac{P(D,A,\neg B,\neg E,F)}{P(A,\neg B,\neg E,F)} = \frac{P(D,A,\neg B,\neg E,F)}{P(D,A,\neg B,\neg E,F) + P(\neg D,A,\neg B,\neg E,F)}$$

$$P(D,A,\neg B,\neg E,F) = P(D,A,\neg B,\neg E,F,C) + P(D,A,\neg B,\neg E,F,\neg C)$$
$$P(\neg D,A,\neg B,\neg E,F) = P(\neg D,A,\neg B,\neg E,F,C) + P(\neg D,A,\neg B,\neg E,F,\neg C)$$

$$P(D,A,\neg B,\neg E,F,C) = P(D|\neg B,C)\cdot P(A)\cdot P(\neg B)\cdot P(\neg E|\neg B,C)\cdot P(F|\neg E)\cdot P(C|A)$$
$$= 0.8\cdot 0.6\cdot 0.8\cdot 0.3\cdot 0.15\cdot 0.4 = 0.006912$$

$$P(D,A,\neg B,\neg E,F,\neg C) = P(D|\neg B,\neg C)\cdot P(A)\cdot P(\neg B)\cdot P(\neg E|\neg B,\neg C)\cdot P(F|\neg E)\cdot P(\neg C|A)$$
$$= 0.7\cdot 0.6\cdot 0.8\cdot 0.75\cdot 0.15\cdot 0.6 = 0.02268$$

$$P(\neg D,A,\neg B,\neg E,F,C) = P(\neg D|\neg B,C)\cdot P(A)\cdot P(\neg B)\cdot P(\neg E|\neg B,C)\cdot P(F|\neg E)\cdot P(C|A)$$
$$= 0.2\cdot 0.6\cdot 0.8\cdot 0.3\cdot 0.15\cdot 0.4 = 0.001728$$

$$\begin{aligned}
P(\neg D, A, \neg B, \neg E, F, \neg C) \\
&= (\neg D|\neg B, \neg C) \cdot P(A) \cdot P(\neg B) \cdot P(\neg E|\neg B, \neg C) \cdot P(F|\neg E) \cdot P(\neg C|A) \\
&= 0.3 \cdot 0.6 \cdot 0.8 \cdot 0.75 \cdot 0.15 \cdot 0.6 = 0.00972
\end{aligned}$$

$$P(D|A, \neg B, \neg E, F) = \frac{0.006912 + 0.02268}{0.006912 + 0.02268 + 0.001728 + 0.00972} = \mathbf{0.721}$$

c) **[0,25 puntos]** Indica las ventajas del método de ponderación de la verosimilitud sobre el muestro estocástico.

Solución: Ambos métodos se utilizan para el cálculo aproximado de probabilidades condicionadas del tipo $P(E_1|E_2)$, dónde E_1 y E_2 son conjuntos de variables aleatorias. No obstante, cuando la probabilidad de que se cumplan los eventos en E_2 es muy baja, el muestreo estocástico genera una gran cantidad de muestras que resultan inútiles para el cálculo de la probabilidad, por lo que se puede considerar una pérdida de tiempo y recursos. El método de ponderación de la verosimilitud corrige este problema, ya que todas las muestras que genera son útiles, en mayor o menor medida, para el cálculo de la probabilidad.

III.- Aprendizaje (3,5 puntos)

5.- **[0,25 puntos]** Explica en que consiste la validación cruzada y sus ventajas y/o desventajas.

Solución: Es una de las opciones para evaluar la capacidad de generalización de un modelo y consiste en dividir el conjunto de instancias en *k* grupos de forma que el aprendizaje se hará *k* veces, cada vez se usarán *k*-1 de los grupos para generar el modelo y el grupo restante para validar. Este sistema permite adaptarse a distintos tamaños de instancias simplemente usando un valor de *k* apropiado. Sin embargo, el tener que repetir *k* veces el ajuste del modelo el coste computacional será alto, especialmente con conjuntos de datos muy grandes.

6.- **[0,75 puntos]** Describe el concepto de sobreajuste y con qué factores de los datos de entrenamiento está relacionado. Explica cómo se puede mitigar en árboles de decisión y SVM.

Solución: Se produce sobreajuste cuando un modelo presenta un ajuste mucho peor en datos de validación (no usados durante el aprendizaje) que la estimación obtenida con los datos de aprendizaje. Por ejemplo, atendiendo al porcentaje de acierto para valorar el ajuste del modelo, si se obtiene un 75% en aprendizaje, pero solo un 50% con datos nuevos hay bastantes indicios de que el modelo sobre ajusta. Uno de los factores que puede contribuir a este fenómeno es el ruido en los datos. Si un modelo intenta ajustar también el ruido en los datos de aprendizaje para maximizar las métricas de evaluación es muy probable que ese modelo dé peor resultado con datos nuevos al tener el ruido un patrón probablemente aleatorio y sobre todo distinto al de los datos sin ruido.
Los árboles de decisión usan el proceso de poda para reducir el número de ramas y evitar que el modelo ajuste detalles muy pequeños que pueden ser ruidosos. Los SVM controlan el efecto negativo del ruido aumentando el número de vectores soporte, mediante el parámetro C, para evitar que un vector soporte ruido tenga mucha influencia en la clasificación.

7.- **[0,5 puntos]** Explica la relación entre las métricas InfoGain (IG) y GainRatio (GR). ¿Hay algún motivo para preferir una sobre la otra?

Solución: La métrica GrainRatio se define como la relación entre InfoGain de la clase y una variable predictora dividido por el SplitInfo de esta variable, que es otro nombre para la entropía de la variable. Con esto, la métrica GR consigue paliar el sesgo de IG hacia variables con muchos valores y que conduciría a modelos con sobreajuste.

8.- **[1 punto]** Considera una red neuronal con **función de activación lineal**. Tiene dos entradas (a, b), una neurona oculta (c) y una unidad de salida (d). Habrá un total de cinco pesos $(w_{ca}, w_{cb}, w_{c0}, w_{dc}, w_{d0})$ que vamos a inicializar todos a 0.1 y con una tasa de aprendizaje $\eta = 0.3$. Calcula el valor de los pesos utilizando el algoritmo de propagación hacía atrás para la siguiente instancia:

a	**b**	**d**
1	1	0

Recuerda que en este caso: $\delta_k = (y_k - o_k)$ y $\delta_h = \sum_k \delta_k w_{kh}$

Solución: $o_c = 0.1 \cdot 1 + 0.1 \cdot 1 + 0.1 \cdot 1 = 0.3$; $o_d = 0.1 \cdot 1 + 0.1 \cdot 0.3 = 0.13$

$\delta_d = 0 - 0.13 = -0.13$; $\delta_c = -0.13 \cdot 0.1 = -0.013$

$$
\begin{aligned}
w_{d0} &= 0.1 + 0.3 \cdot (-0.13) \cdot 1 = 0.061 \\
w_{dc} &= 0.1 + 0.3 \cdot (-0.13) \cdot 0.3 = 0.0883 \\
w_{c0} &= 0.1 + 0.3 \cdot (-0.013) \cdot 1 = 0.0961 \\
w_{ca} &= 0.1 + 0.3 \cdot (-0.013) \cdot 1 = 0.0961 \\
w_{cb} &= 0.1 + 0.3 \cdot (-0.013) \cdot 1 = 0.0961
\end{aligned}
$$

9.- **[1 punto]** Considera el siguiente conjunto de entrenamiento, al que se le ha aplicado SVM con kernel lineal para predecir la variable Y. Sabiendo que $b = 10$, calcula la predicción de este modelo para la instancia X1=3, X2=8.

X1	4	5	9	7	2	4	9	2
X2	5	6	4	9	1	4	8	0
Y	1	-1	-1	-1	1	1	-1	1
α	1	1	0	0	0	0	0	0

Solución:

$$
g\left(\begin{pmatrix}3\\8\end{pmatrix}\right) = \sum_{i \in VS} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b = 1 \cdot 1 \cdot \begin{pmatrix}4\\5\end{pmatrix}^T \cdot \begin{pmatrix}3\\8\end{pmatrix} + 1 \cdot (-1) \cdot \begin{pmatrix}5\\6\end{pmatrix}^T \cdot \begin{pmatrix}3\\8\end{pmatrix} + 10
$$

$$
= 52 - 63 + 10 = -1
$$

Alternativa, calculando $\boldsymbol{w}$:

$$
\boldsymbol{w} = \sum_{i \in VS} \alpha_i y_i \boldsymbol{x}_i = 1 \cdot 1 \cdot \begin{pmatrix}4\\5\end{pmatrix} + 1 \cdot (-1) \cdot \begin{pmatrix}5\\6\end{pmatrix} = \begin{pmatrix}-1\\-1\end{pmatrix}
$$

$$
g\left(\begin{pmatrix}3\\8\end{pmatrix}\right) = \boldsymbol{wx} + b = \begin{pmatrix}-1\\-1\end{pmatrix}^T \cdot \begin{pmatrix}3\\8\end{pmatrix} + 10 = -1
$$

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_d \sum_k (y_{kd} - o_{kd})^2$$

$$b = \frac{1}{|VS|} \sum_{i \in VS} y_i - \boldsymbol{wx_i}$$

$$d = \frac{2}{\|\boldsymbol{w}\|}$$

$$K(\boldsymbol{x}, \boldsymbol{y}) = e^{-\|x-y\|^2/(2\sigma^2)}$$

$$H(X|Y) = \sum_{y \in Y} p(y) H(X|Y = y)$$

$$H(X) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

$$K(x_i, x_j) = \phi(x_i)\phi(x_j)$$

$$g(\cdot) = \begin{cases} 1 \text{ si } \boldsymbol{wx}>0 \\ -1 \text{ en otro caso} \end{cases}$$

$$\Delta w_i = \eta(y_d - \boldsymbol{wx}_d) x_{di}$$

$$dist(\vec{x}, \vec{y}) = \left(\sum_i |x_i - y_i|^p\right)^{1/p}$$

$$\boldsymbol{w} = \sum_{i \in VS} \alpha_i y_i \boldsymbol{x_i}$$

$$K(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}\boldsymbol{y} + 1)^d$$

$$E(\boldsymbol{w}) = \tfrac{1}{2}(y_d - \boldsymbol{w}\boldsymbol{x}_d)^2$$

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

$$y(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} \; + b$$

$$g(\boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x} + b$$